GeminiDB

Mongo

Issue 24

Date 2024-07-30

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Service Overview

1.1 What Is a GeminiDB Mongo API?

GeminiDB Mongo API is a cloud-native NoSQL high availability database compatible with MongoDB. This flexible, scalable, and reliable database can be managed on a visualized platform.

- Enterprise-class performance

  GeminiDB Mongo API gives you 3 times the performance of the open source version. Data can be written to this high availability database 24/7, and with automated load balancing and elastic scaling, you always have all the performance you need.
- Flexibility

  The distributed architecture, with decoupled compute and storage, means that compute nodes can be expanded in minutes, and storage capacity can be scaled up in seconds without affecting your services.
- Ultimate reliability
  - A multi-layer security system, including VPC, subnet, security group, SSL, and fine-grained permission control, is used to ensure database security and user privacy.
  - You can deploy nodes across three AZs and quickly back up or restore data to improve data reliability.
  - The distributed architecture provides superlative fault tolerance (*N-1* reliability).
- Friendly UI

  On the visualized instance management console, you can create or delete instances in just a few clicks. Backup and restoration, configuring alarms, and adding or removing nodes is just as easy.

Typical Application Scenarios

Gaming

GeminiDB Mongo API is compatible with MongoDB, so you can store game data, such as user equipment and user bonus points in the database. The number of

compute nodes in GeminiDB Mongo instances can be dynamically increased, making it ideal for high-concurrency scenarios of gaming.

Advantages

- Flexibility

  Generally, you need to flexibly increase compute nodes within 6 hours of game service provisioning to handle performance challenges. GeminiDB Mongo API is well suited for this scenario.

- Fast recovery

  GeminiDB Mongo API supports table-level, point-in-time recovery, so you can restore game data quickly.

- Stable expansion

  You can scale up capacity without impacting the experience for ongoing games.

1.2 Compatible APIs and Versions

This section describes compatible APIs and versions supported by GeminiDB Mongo API.

Table 1-1 Compatible APIs and versions

Compatible API	Instance Type	Version
MongoDB	Replica set	4.0

1.3 Instance Specifications

Instances of the same type can have different memory specifications. You can select instances of different specifications based on application scenarios.

This section describes the instance specifications supported by GeminiDB Mongo API. The instance specifications depend on the selected CPU model.

GeminiDB Mongo supports replica set instances of different specifications.

Replica Set Instance Specifications

Table 1-2 GeminiDB Mongo replica set instance specifications

Flavor	vCPUs	Memory (GB)	Min. Storage Space (GB)	Max. Storage Space (GB)	Default Maximum Connections
geminidb.mongodb.repset.large.4	2	8	100	600	400

Flavor	vCPUs	Memory (GB)	Min. Storage Space (GB)	Max. Storage Space (GB)	Default Maximum Connections
geminidb.mongodb.repset.xlarge.4	4	16	100	1,200	1,000
geminidb.mongodb.repset.2xlarge.4	8	32	100	2,400	4,000
geminidb.mongodb.repset.4xlarge.4	16	64	100	4,800	8,000
geminidb.mongodb.repset.8xlarge.4	32	128	100	9,600	16,000
geminidb.mongodb.repset.xlarge.8	4	32	100	1,200	1,000
geminidb.mongodb.repset.2xlarge.8	8	64	100	2,400	4,000
geminidb.mongodb.repset.4xlarge.8	16	128	100	4,800	8,000
geminidb.mongodb.repset.8xlarge.8	32	256	100	9,600	16,000

1.4 DB Instance Statuses

The status of a DB instance indicates the health of the instance. You can view the DB instance statuses on the management console.

Table 1-3 DB instance statuses

Status	Description
Available	DB instance is available.
Abnormal	DB instance is faulty.
Creating	DB instance is being created.
Creation failed	DB instance creation fails.
Restarting	DB instance is being restarted.
Resetting password	Administrator password is being reset.
Scaling up	The storage space of the DB instance is being expanded.

Status	Description
Changing instance class	The CPU or memory of a DB instance is being changed.
Uploading backup	The backup file is being uploaded.
Backing up	Backup is being created.
Checking restoration	The backup of the current DB instance is being restored to a new DB instance.
Configuring SSL	SSL is being enabled or disabled.
Changing to yearly/monthly	The billing mode is being changed from pay-per-use to yearly/monthly.
Changing to pay-per-use	The billing mode is being changed from yearly/monthly to pay-per-use.
Frozen	The instance is frozen because your balance drops to or below zero.
Unfreezing	DB instance is unfrozen after the overdue payments are cleared.
Checking changes	The yearly/monthly instance is pending check when its billing mode is changed.

1.5 Usage Specifications

1.5.1 Naming Rules

- It is recommended that the name of a database object (database, table, field, or index) starts with a lowercase letter followed by letters and digits. The length of the name cannot exceed 32 bytes.
- The database name cannot contain special characters (""./\$/*?~#:?|") and null character (\0). The database name cannot be the system database name, such as admin, local, and config.
- The database collection name can only contain letters and underscores (_). The name cannot be prefixed with "system". The total length of <*Database name*>.<*Collection name*>. cannot exceed 120 characters.

1.5.2 Indexing Rules

- A column index can have up to 512 bytes, an index name can have up to 64 characters, and a composite index can have up to 16 columns.
- The total length of <*Database name*>.<*Collection name*>.$<*Index name*> cannot exceed 128 characters.
- Use indexes with high selectivity. Indexes with low selectivity will return large number of rows for each key value.

- Write operations on a collection trigger more I/O operations on indexes in the collection. Ensure that the number of indexes in the collection does not exceed 32.
- Do not create indexes that will not be used. Unused indexes loaded to the memory will cause a waste of memory. In addition, useless indexes generated due to changes in service logic must be deleted in a timely manner.
- Indexes must be created in the background instead of foreground.
- An index must be created for the sort key. If a composite index is created, the column sequence of the index must be the same as that of the sort key. Otherwise, the index will not be used.
- Do not create an index based on the leading-edge column of a composite index. If the leading-edge column of a composite index is the column used in another index, the smaller index can be removed. For example, a composite index based on "firstname" and "lastname" can be used for queries on "firstname". In this case, the firstname-based index is unnecessary.
- The time required for creating an index depends on the data volume. You are advised to design the index in advance.
- You can run the **db.currentOp** command and view the index creation progress in the command output (keyword: **createIndexes**).

1.5.3 Database Connection

When the maximum number of mongod connections is reached, your client cannot connect to the GeminiDB Mongo instances. In the mongod service model, each network connection uses a single thread of 1 MB stack space. As the network connections increase, too many threads will increase the context switch overhead and memory usage.

- When your clients use the GeminiDB Mongo driver to connect to databases, a connection pool must be configured. The maximum number of connections in a connection pool cannot exceed 200.
- When your clients use the GeminiDB Mongo driver to connect to databases, calculate the number of clients and the size of the connection pool configured for each client. The total number of connections cannot exceed 80% of the maximum number of connections.
- For a replica set, configure the IP addresses of both the primary and secondary nodes on the client.
- GeminiDB Mongo provides the **rwuser** user by default. The authentication database must be admin when the **rwuser** user is used for login.

1.5.4 Reliability

- Rules for setting write concern:

  For mission-critical services, set write concern to {w:n},n>0. A larger value indicates better consistency but poorer performance.

  - **w:1** indicates that a confirmation message is returned after data is written to the primary node.
  - **w:1,journal:true** indicates that the result is returned after data is written to the primary node and logs.

  - **w:majority** indicates that the result is returned after data is written to more than half of the total nodes.

    NOTE

    If data is not written using **w:majority**, the data that is not synchronized to the standby node may be lost when a primary/standby switchover occurs.

- If high reliability is required, deploy a replica set across three AZs.

1.5.5 Database Performance

- Full table scan is not allowed.
- During the query, select only the fields that need to be returned. In this way, the network and thread processing loads are reduced. If you need to modify data, modify only the fields that need to be modified. Do not directly modify the entire object.
- Do not use $not. GeminiDB Mongo does not index missing data. The $not query requires that all records be scanned in a single result collection. If $not is the only query condition, a full table scan will be performed on the collection.
- If you use $and, put the conditions with the fewest matches before other conditions. If you use $or, put the conditions with the more matches first.
- In a single instance, the total number of databases cannot exceed 200, and the total number of collections cannot exceed 500. If the number of collections is too large, the memory may be overloaded. In addition, the performance may deteriorate when a device restarts or an active/standby switchover is triggered, which affects the high availability performance in emergencies.
- Before bringing a service online, perform a load test to measure the performance of the database in peak hours.
- Do not execute a large number of concurrent transactions at the same time or leave a transaction uncommitted for a long time.
- Before rolling out services, check the performance of all query types through the execution of query plans.

  During the development process, each execution on a collection must be checked using explain() to view its execution plan.

  Example:

  db.T_DeviceData.find({"deviceId" : "ae4b5769-896f"}).explain();

  db.T_DeviceData.find({"deviceId" : "77557c2-31b4"}).explain("executionStats");

  A covered query does not have to read a document and returns a result from an index, so using a covered query can greatly improve query efficiency. If the output of explain() shows that **indexOnly** is **true**, the query is covered by an index.

  Execution plan parsing:

  - Check the execution time. The smaller the values of the following parameters, the better the performance: **executionStats.executionStages.executionTimeMillisEstimate** and **executionStats.executionStages.inputStage.executionTimeMillisEstimate**

- **executionStats.executionTimeMillis** specifies how much time the database took to both select and execute the winning plan.
- **executionStats.executionStages.executionTimeMillisEstimate** indicates the execution completion time of the winning plan.
- executionStats.executionStages.inputStage. **executionTimeMillisEstimate** indicates the execution completion time of the child stage of the winning plan.

– Check the number of scanned records. If the three items are the same, the index is best used.

- **executionStats. nReturned** indicates the number of documents that match the query condition.
- **executionStats .totalKeysExamined** indicates the number of scanned index entries.
- **executionStats .totalDocsExamined** indicates the number of scanned document entries.

– Check the stage status (combination of stages with good performance)

- Fetch+IDHACK
- Fetch+ixscan
- Limit+ (Fetch+ixscan)
- PROJECTION+ixscan

Table 1-4 Status description

Status Name	Description
COLLSCAN	Full Table Scan
SORT	In-memory sorting
IDHACK	_id-based query
TEXT	Full-text index
COUNTSCAN	Number of unused indexes
FETCH	Index scanning
LIMIT	Using Limit to limit the number of returned records
SUBPLA	$or query stage without using an index
PROJECTION	Number of used indexes
COUNT_SCAN	Number of used indexes

1.5.6 Cursor Usage Rules

By default, the server automatically closes the cursor that has no activity in the past 10 minutes or the one whose associated result sets are all queried. If you do not need a cursor any more, close it immediately to save system resources.

2 Billing

2.1 Billing Overview

In this document, you will learn about how instances are billed, how you can renew subscriptions and manage costs, and what happens if your account goes into arrears.

- **Billing Modes**

  There are yearly/monthly and pay-per-use billing modes. Each one has different advantages and disadvantages.

  – Yearly/Monthly: You pay upfront for the amount of time you expect to use the instance for. You will need to make sure you have a top-up account with a sufficient balance or have a valid payment method configured first.

  – Pay-per-use: You can start using the GeminiDB instance first and then pay as you go.

  For details about the two billing modes, see **Overview**.

  You can also change the billing mode later if it no longer meets your needs. For details, see **Overview**.

- **Billing Items**

  You will be billed for instance specifications, storage space, backup space, and EIP bandwidths. For details about the billing factors and formulas for each billed item, see **Billing Items**.

  For more information about billing samples and the billing for each item, see **Billing Examples**.

- **Renewing Subscriptions**

  If you want to continue using an instance after it expires, you need to renew the instance subscription within the specified period. Otherwise, resources, such as compute and storage, will be automatically released, and data may be lost.

  You can renew your subscription manually or automatically. For details, see **Overview**.

- **Viewing Bills**

You can choose **Billing & Costs** > **Bills** to check the instance transactions and bills. For details, see **Bills**.

- **Arrears**

  If there is not a sufficient account balance to pay for your bill and there is no other payment method configured, your account will go into arrears. If you want to continue using your cloud services, you will need to top up your account in a timely manner. For details, see **Arrears**.

- **Stopping Billing**

  If you no longer need to use your instance, you can unsubscribe from or delete it to stop the billing. For details, see **Billing Termination**.

- **Managing Costs**

  GeminiDB Mongo costs include resource costs and O&M costs. You can allocate, analyze, and optimize GeminiDB costs to save more money. For details, see **Cost Management**.

2.2 Billing Modes

2.2.1 Overview

There are yearly/monthly and pay-per-use billing modes. Each one has different advantages and disadvantages.

- Yearly/Monthly is a prepaid billing mode. You pay in advance for a subscription term, and in exchange, you get a discounted rate. The longer the subscription term, the bigger the discount. Yearly/Monthly billing is a good option for long-term, stable services.
- Pay-per-use is a postpaid billing mode. You pay as you go and just pay for what you use. The instance usage is calculated by the second but billed every hour. Pay-per-use billing is a good option for scenarios where there are sudden traffic bursts, such as e-commerce promotions.

Table 2-1 lists differences between the two billing modes.

Table 2-1 Differences between billing modes

Billing Mode	Yearly/Monthly	Pay-per-use
Payment	Prepaid Billed by the subscription term you purchase	Postpaid Billed for what you use
Billing Method	Billed by the subscription term you purchase	Calculated by the second but billed every hour
Billed Items	Instance specifications (vCPUs and memory), storage space, backup space, and EIPs	Instance specifications (vCPUs and memory), storage space, backup space, and EIPs

Changing the Billing Mode	Yearly/Monthly can be changed to pay-per-use. The change takes effect only after the yearly/monthly subscription expires. For details, see **Yearly/Monthly to Pay-per-Use**.	Pay-per-use can be changed to yearly/monthly. For details, see **Pay-per-Use to Yearly/Monthly**.
Changing the Specifications	Supported	Supported
Application Scenarios	Recommended for resources expected to be in use long term. A cost-effective option for scenarios where the resource usage duration is predictable.	Recommended when the resource demands are likely to fluctuate and you want more flexibility.

2.2.2 Yearly/Monthly Billing

If you expect to use resources for a longer period, you can save money by selecting yearly/monthly billing. This section describes billing rules for yearly/monthly GeminiDB Mongo resources.

Application Scenarios

If you want to ensure resource stability over a certain period of time, yearly/monthly billing is a good choice for the following types of workloads:

- Long-term workloads with stable resource requirements, such as official websites, online malls, and blogs.
- Long-term projects, such as scientific research projects and large-scale events.
- Workloads with predictable traffic bursts, for example, e-commerce promotions or festivals.
- Workloads with high data security requirements.

Billed Items

You are billed for the following items on a yearly/monthly basis.

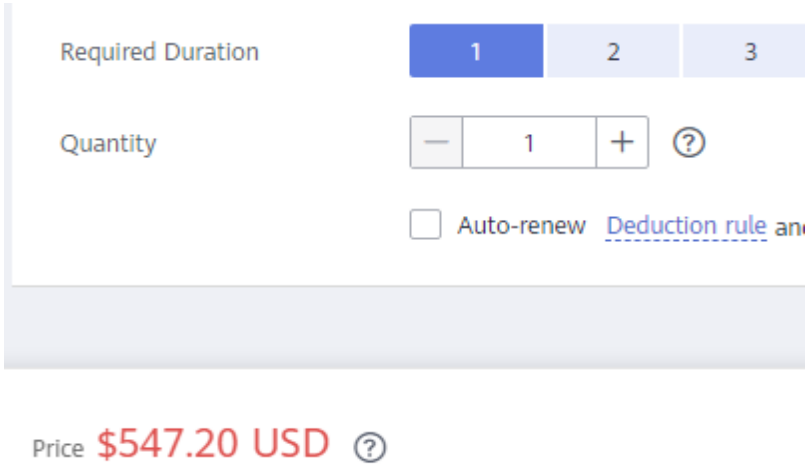
Table 2-2 Items billed on a yearly/monthly basis

Billed Item	Description
Instance specifications	Instance specifications, including vCPUs and memory.

Billed Item	Description
Storage space	If the actual storage usage exceeds your purchased storage, you will be billed for additional storage on a pay-per-use basis.
Backup space	GeminiDB Mongo provides backup storage up to 100% of the database storage you purchase at no additional charge. After the free backup space is used up, charges are applied based on the backup space pricing details. Pricing is listed on a per-hour basis, but bills are calculated based on the actual usage duration.
(Optional) Public network bandwidth	GeminiDB Mongo instances are accessible from public networks, and you are billed for the generated public network traffic, but not for private network traffic.

If you want to purchase a 3-node (specifications of each node: 4 vCPUs | 16 GB) GeminiDB Mongo instance with 12 GB of storage space. At the bottom of the instance buying page, price details (excluding the backup space fee) will be displayed.

Figure 2-1 Example price



The price includes:

- Selected specifications for your instance
- Storage space

NOTE

The backup space fee is not included. For details about the backup price, see **Product Pricing Details**.

Backup Storage Space

DB Instance Type	Hourly	Currency
Cluster	0.00004	Price per GB

Billed Usage Period

A yearly/monthly instance is billed for the purchased duration (UTC+8). The billing starts when you activated or renewed the subscription, and ends at 23:59:59 of the expiry date.

For example, if you purchased a one-month GeminiDB Mongo instance on March 08, 2023, 15:50:04, the billed usage period is from March 08, 2023, 15:50:04 to April 08, 2023, 23:59:59.

Billing Examples

Suppose you purchased a one-month GeminiDB Mongo instance (instance specifications: 4 vCPUs | 16 GB; nodes: 3; storage: 100 GB; backup space: 110 GB (100 GB for free)) on March 08, 2023, 15:50:04, and renewed the subscription for one more month before the initial subscription expired. That would include two usage periods:

- March 08, 2023, 15:50:04 to April 08, 2023, 23:59:59
- April 08, 2023, 23:59:59 to May 08, 2023, 23:59:59
  - From April 08, 2023, 23:59:59 to May 01, 2023, 23:59:59, 50 GB of free backup space was used.
  - From May 01, 2023, 23:59:59 to May 08, 2023, 23:59:59, another 10 GB of backup space was used, which was billed for 168 hours.

You will be billed for both usage periods. GeminiDB Mongo resources are billed individually as follows.

Table 2-3 Formulas for billing yearly/monthly resources

Resource	Formula	Unit Price
Instance specifications (including vCPUs and memory)	Unit price of the instance specifications x Required duration x Number of nodes	For details about the unit price, see **Cluster CPU/Memory** on **Product Pricing Details**.
Storage space	Storage space unit price x Required duration x Storage space (GB)	For details about the unit price, see **Storage Space** on **Product Pricing Details**.

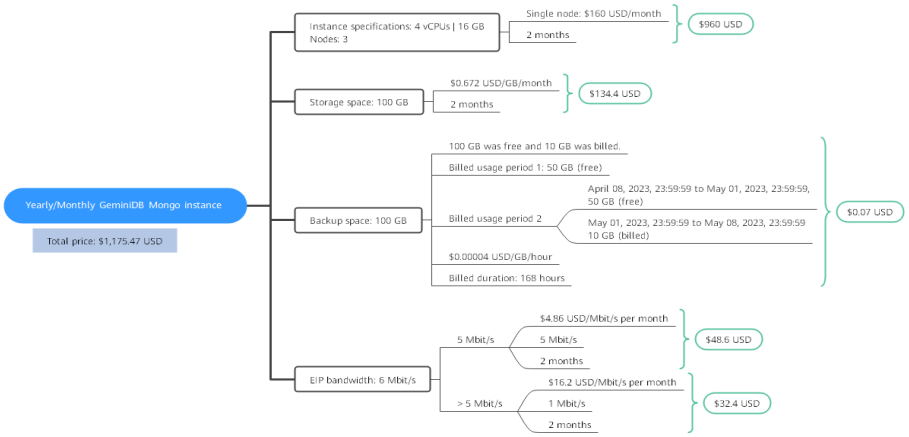
Resource	Formula	Unit Price
Backup space	Backup space unit price x Required duration x (Backup space – Storage space) (GB) **NOTE** The billed duration refers to the length of time the billed backup space was used for.	For details about the unit price, see **Backup Storage Space** on **Product Pricing Details**.
Public network bandwidth	Billed by fixed bandwidth	For details, see **Product Pricing Details**.

Figure 2-2 shows how the total price is calculated.

NOTICE

Prices in the figure are just examples. Actual prices are subject to **Product Pricing Details**.

Figure 2-2 Total price for a yearly/monthly GeminiDB Mongo instance



Price Change After Specification Change

If the specifications of a yearly/monthly GeminiDB Mongo instance no longer meet your needs, you can change the specifications on the console. The system will recalculate the price and either bill or refund you the difference.

- If you upgrade your GeminiDB Mongo instance specifications, you need to pay the difference in price.
- If you downgrade your GeminiDB Mongo instance specifications, Huawei Cloud will refund you the difference.

You are not advised to downgrade your GeminiDB Mongo instance to a lower specification because the instance performance may be affected. Suppose you

purchased a yearly/monthly GeminiDB Mongo instance (4 vCPUs | 16 GB and 3 nodes) on April 08, 2023 and upgraded the instance specifications to 8 vCPUs | 32 GB and 3 nodes on April 18, 2023. The price for the original specifications was $547.20 USD/month, and that for the new specifications was $1,027.20 USD/month. The price difference will be calculated as follows:

Price difference for the specification upgrade = Price for the new specifications × Remaining period - Price for the original specifications × Remaining period

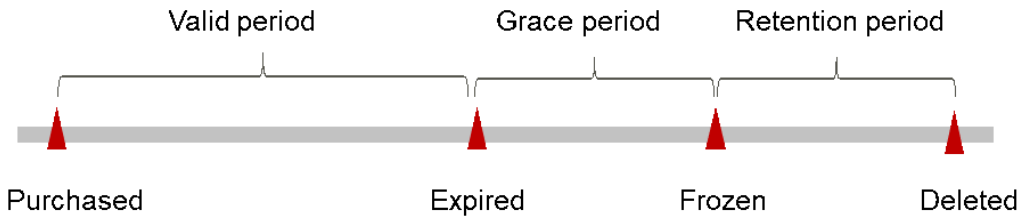
The remaining period in the formula is the remaining days of each calendar month divided by the maximum number of days in each calendar month. In this example, Remaining period = 12 (Remaining days in April)/30 (Maximum number of days in April) + 8 (Remaining days in May)/31 (Maximum number of days in May) = 0.6581. Cost of upgrade = $1,027.20 USD × 0.6581 – $547.20 USD × 0.6581 = $315.89 USD.

For more details, see **Pricing of a Changed Specification**.

Impact of Expiration

Figure 2-3 shows the statuses a yearly/monthly GeminiDB Mongo instance can go through throughout its lifecycle. After a GeminiDB Mongo instance is purchased, it enters the valid period and runs normally during this period. If the instance is not renewed after it expires, before being deleted, it first enters a grace period and then a retention period.

Figure 2-3 Lifecycle of a yearly/monthly GeminiDB Mongo instance



Expiration Reminder

The system will send you a reminder (by email, SMS, or in-app message) 7 days before a yearly/monthly GeminiDB Mongo instance expires to remind you to renew the subscription.

Impact of Expiration

If your yearly/monthly GeminiDB Mongo instance is not renewed after it expires, it changes to the **Expired** state and enters a grace period. During the grace period, you can access the GeminiDB Mongo instance but cannot:

- Change instance specifications.
- Change the billing mode from yearly/monthly to pay-per-use.
- Unsubscribe from it.

If the yearly/monthly GeminiDB Mongo instance is not renewed after the grace period ends, its status turns to **Frozen** and it enters a retention period. You cannot

perform any operations on the GeminiDB Mongo instance while it is in the retention period.

If the yearly/monthly GeminiDB Mongo instance is not renewed by the time the retention period ends, it will be released and data cannot be restored.

NOTE

- For details about renewals, see **Overview**.

2.2.3 Pay-per-Use Billing

Pay-per-use billing means you pay nothing up front and are not tied into any contract or commitment. This section describes billing rules for pay-per-use GeminiDB Mongo instances.

Application Scenarios

Pay-per-use billing is good for short-term, bursty, or unpredictable workloads that cannot tolerate any interruptions, such as applications for e-commerce flash sales, temporary testing, and scientific computing.

Billing Items

You are billed for the following items on a pay-per-use basis.

Table 2-4 Items billed on a pay-per-use basis

Billing Item	Description
Instance specifications	Instance specifications, including vCPUs and memory.
Storage space	Instance storage space, which is billed hourly on a pay-per-use basis.
Backup space	GeminiDB Mongo provides backup storage up to 100% of the database storage you purchase at no additional charge. After the free backup space is used up, charges are applied based on the backup space pricing details. Pricing is listed on a per-hour basis, but bills are calculated based on the actual usage duration.
(Optional) Public network bandwidth	GeminiDB Mongo instances are accessible from public networks, and you are billed for the generated public network traffic, but not for private network traffic.

If you want to purchase a 3-node (specifications of each node: 4 vCPUs | 16 GB) GeminiDB Mongo instance with 100 GB of storage space. At the bottom of the instance buying page, price details (excluding the backup space fee) will be displayed.

Figure 2-4 Example price

The price includes:

- Instance specifications (including vCPUs and memory)
- Selected storage space

NOTE

The backup space fee is not included. For details about the backup price, see **Product Pricing Details**.

Backup Storage Space

DB Instance Type	Hourly	Currency
Cluster	0.00004	Price per GB

Billed Usage Period

Pay-per-use GeminiDB Mongo instance usage is calculated by the second and billed every hour. The billing starts when ECS instance is created and ends when the instance is deleted.

NOTE

It takes a certain time to create an instance. The billing starts from the time when the instance is successfully created. You can view the two time points on the **Basic Information** page. You can view the time when the instance is created beside the **Created** field.

For example, if you purchased a pay-per-use GeminiDB Mongo instance at 8:45:30 and deleted it at 8:55:30, you are billed for the 600 seconds from 8:45:30 to 8:55:30.

Billing Examples

Suppose you purchased a pay-per-use instance on April 18, 2023, 9:59:30, and deleted it on April 18, 2023, 10:45:46. Two usage periods will be billed:

- Usage of 30 seconds from 9:59:30 to 10:00:00
- Usage of 2,746 seconds from 10:00:00 to 10:45:46
  - The free backup space is used from 10:00:00 to 10:45:00.
  - Ten GB of billing backup space is used from 10:45:00 to 10:45:46 and the billed duration is 46 seconds.

The price displayed in the pricing details is per hour, so you need to divide it by 3,600 to obtain the price for each second and then multiply the per-second price by the total number of seconds. GeminiDB Mongo instances are billed individually as follows.

Table 2-5 Formulas for billing pay-per-use resources

Resource	Formula	Unit Price
Compute resources (including vCPUs and nodes)	Unit price of instance specifications x Required duration	For details about the unit price, see **Cluster CPU/Memory** on **Product Pricing Details**
Storage space	Storage space unit price x Purchase duration	For details about the unit price, see **Storage Space** on **Product Pricing Details**.
Backup space	Backup space unit price x Required duration x (Backup space – Storage space) (GB) **NOTE** The billed duration refers to the length of time the billed backup space was used for.	For details about the unit price, see **Backup Storage Space** on **Product Pricing Details**.
Public network traffic	Tiered billing by fixed bandwidth • 0 Mbit/s to 5 Mbit/s (included): billed at a fixed unit price per Mbit/s • Greater than 5 Mbit/s: billed at a different price per Mbit/s	For details, see **Bandwidth Price** on **Product Pricing Details** page or **Product Pricing Details**.

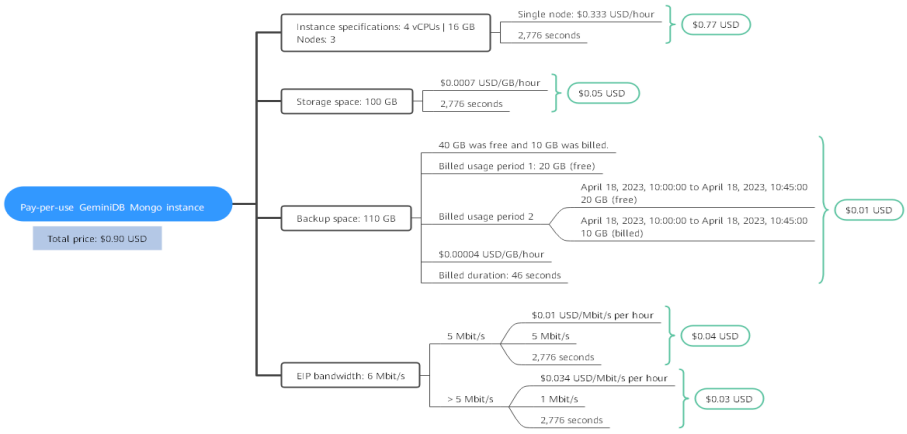
Figure 2-5 shows how the total price is calculated.

NOTICE

Prices in the figure are just examples. Actual prices are subject to **Product Pricing Details**.

For pay-per-use billing, decimal numerals on the price calculator are rounded off and are accurate to two decimal places. If the fee is less than $0.01 USD (after rounding off), $0.01 USD will be displayed.

Figure 2-5 Total price for a pay-per-use GeminiDB Mongo instance



Price Change After Specification Change

If you change the specifications of a pay-per-use GeminiDB Mongo instance, the original order will become invalid and a new order will be placed. You will be billed based on the new specifications.

If you change instance specifications within a given hour, multiple records will be generated. Different records record the billing for different specifications.

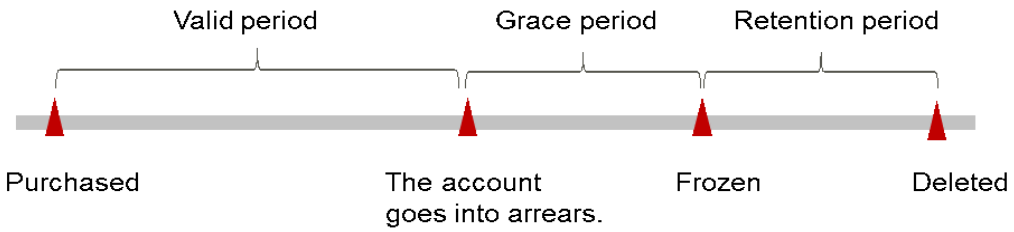
For example, if you purchased a pay-per-use instance (4 vCPUs | 16 GB) at 9:00:00 and changed the instance specifications to 8 vCPUs | 32 GB at 9:30:00, the following items will be billed:

- Specifications 4 vCPUs | 16 GB usage from 9:00:00 to 9:30:00
- Specifications 8 vCPUs | 32 GB usage from 9:30:00 to 10:00:00

Impacts of Arrears

Figure 2-6 shows the statuses a pay-per-use GeminiDB Mongo instance can go through throughout its lifecycle. After a GeminiDB Mongo instance is purchased, it enters the valid period and runs normally during this period. If your account goes into arrears, the instance enters a grace period and then a retention period.

Figure 2-6 Lifecycle of a pay-per-use GeminiDB Mongo instance



Arrears Reminder

The system will bill you for pay-per-use resources after each billing cycle ends. If your account goes into arrears, we will notify you by email, SMS, or internal message.

Impacts of Arrears

When your account is in arrears due to automatic fee deduction for pay-per-use GeminiDB Mongo instances, the account status turns to arrears. In arrears, the pay-per-use instance continues rendering service but the instance enters the grace period. You are still responsible for expenditures generated during the grace period. You can view the charges on the **Billing Center** > **Overview** page and pay any past due balance as needed.

If you do not bring your account balance current before the grace period expires, the GeminiDB Mongo instance status turns to **Frozen** and it enters a retention period. You cannot perform any operations on a pay-per-use GeminiDB Mongo instance in the **Frozen** status.

If you do not bring your account balance current before the retention period ends, your instance will be released, and data cannot be restored.

NOTE

- During the retention period, you cannot access or use your instance but the data stored in it can be retained. The retention period for Huawei Cloud International website is 15 days.
- During the grace period, you can access and use only some resources of your instance. The grace period for Huawei Cloud International website is 15 days.
- For details about top-up, see **Topping Up an Account**.

2.3 Billing Items

Billing

You will be billed for instance specifications, storage space, backup space, and public network traffic. For details, see **Table 2-6**.

NOTE

The billed items marked with asterisks (*) are mandatory.

Table 2-6 Billing items of a GeminiDB Mongo instance

Billing Item	Description	Billing Mode	Formula
* Specifications	Billed by instance specifications, including vCPUs and memory. Computing and storage capabilities vary by the number of vCPUs and memory size.	Yearly/ Monthly and pay-per-use	**Unit price x Required duration** For details about the unit price, see **Cluster CPU/Memory** on **Product Pricing Details**.

Billing Item	Description	Billing Mode	Formula
* Storage space	Billed based on unified standards.	Yearly/ Monthly and pay-per-use	**Unit price x Storage space x Required duration** For details about the unit price, see **Storage Space** on **Product Pricing Details**.
Backup space	Billed based on unified standards.	Pay-per-use	**Unit price x Billed backup space x Required duration** For details about the unit price, see **Backup Storage Space** on **Product Pricing Details**. **NOTE** The billed duration refers to the length of time the billed backup space was used for.
Public network traffic	An EIP is required if a GeminiDB Mongo instance needs to access the Internet. Billed by bandwidth, traffic, and the EIP reservation price. • EIP for a yearly/monthly GeminiDB Mongo instance: billed by bandwidth. • EIP for a pay-per-use GeminiDB Mongo instance: billed by bandwidth, traffic, or shared bandwidth. You are also charged for IP reservation if you do not bind the EIP to any instance.	Yearly/ Monthly and pay-per-use You can purchase a bandwidth add-on package or a shared traffic package.	**Tiered pricing based on fixed bandwidth**. • 0 Mbit/s to 5 Mbit/s (included): billed at a fixed unit price per Mbit/s. • Greater than 5 Mbit/s: billed at a different price per Mbit/s. For details about the unit price, see **Bandwidth Price** on **Product Pricing Details** or **Product Pricing Details**.

Billing Examples

Suppose you purchased a one-month GeminiDB Mongo instance (instance specifications: 4 vCPUs | 16 GB; nodes: 3; storage: 100 GB; backup space: 110 GB (100 GB for free)) on March 08, 2023, 15:50:04, and renewed the subscription for one more month before the initial subscription expired. That would include two usage periods:

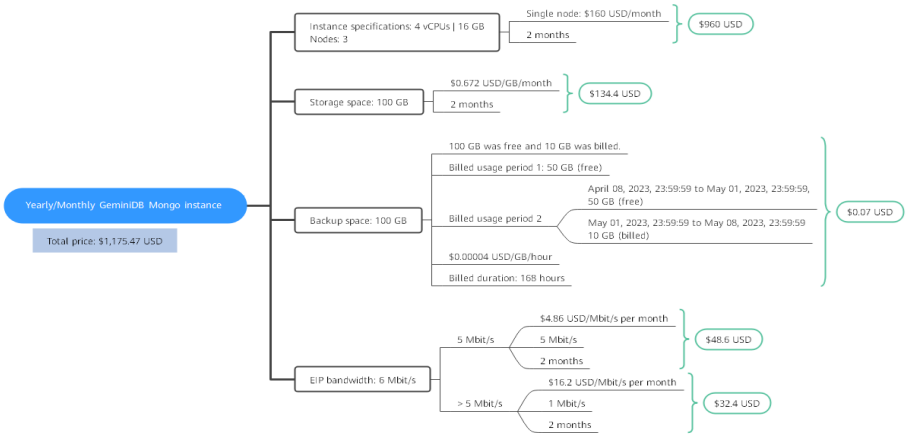
- March 08, 2023, 15:50:04 to April 08, 2023, 23:59:59
- April 08, 2023, 23:59:59 to May 08, 2023, 23:59:59
  - From April 08, 2023, 23:59:59 to May 01, 2023, 23:59:59, 50 GB of free backup space was used.
  - From May 01, 2023, 23:59:59 to May 08, 2023, 23:59:59, another 10 GB of backup space was used, which was billed for 168 hours.

Figure 2-7 shows how the total price is calculated.

NOTICE

Prices in the figure are only for reference. For details, see **Product Pricing Details**.

Figure 2-7 Total price for a yearly/monthly GeminiDB Mongo instance



For more billing examples of a pay-per-use GeminiDB Mongo instance, see **Billing Examples**.

2.4 Billing Examples

Billing Scenario

A user purchased a pay-per-use GeminiDB Mongo instance at 15:30:00 on March 18, 2023. The instance configuration is as follows:

- Specifications: 4 vCPUs | 16 GB
- Nodes: 3
- Public network bandwidth: 6 Mbit/s

After a period of time, the user found that the current GeminiDB Influx instance specifications no longer met service requirements and updated the specifications to 8 vCPUs | 32 GB at 09:00:00 on March 20, 2023. Since the user wanted to use the instance long term, the user then changed the instance to yearly/monthly billing with a one-month duration at 10:30:00 on the same day. So how much will the user be billed for this GeminiDB Influx instance in March and April?

Billing Analysis

The total price of this GeminiDB Mongo instance involves both pay-per-use and yearly/monthly usage:

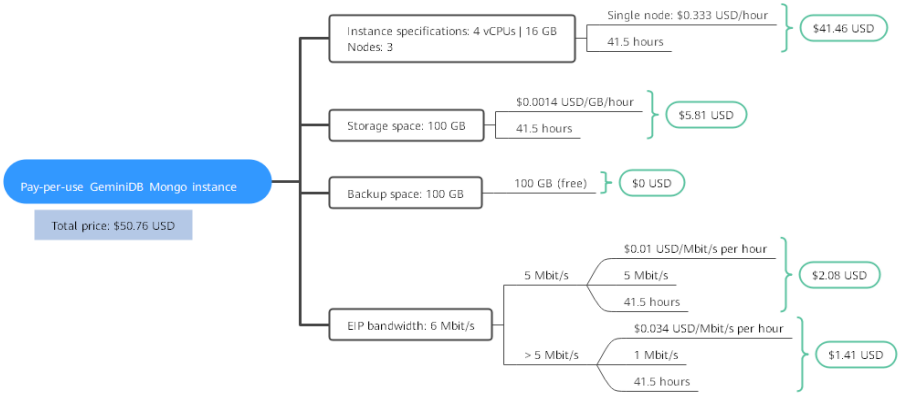
- Pay-per-use usage: March 18, 2023, 15:30:00 to March 20, 2023, 10:30:00
  - March 18, 2023, 15:30:00 to March 20, 2023, 9:00:00
    - Instance specifications: 4 vCPUs | 16 GB
    - Nodes: 3
    - Used storage space: 100 GB
    - Used backup space: 100 GB
    - Public network bandwidth: 6 Mbit/s
  - March 20, 2023, 9:00:00 to March 20, 2023, 10:30:00
    - Instance specifications: 8 vCPUs | 32 GB
    - Nodes: 3
    - Used storage space: 200 GB
    - Used backup space: 210 GB (billed on a pay-per-use basis from March 20, 2023, 10:00:00 to March 20, 2023, 10:30:00)
    - Public network bandwidth: 6 Mbit/s
- Yearly/Monthly: March 20, 2023, 10:30:00 to April 20, 2023, 23:59:59
  - Instance specifications: 8 vCPUs | 32 GB
  - Nodes: 3
  - Used storage space: 200 GB
  - Used backup space: 300 GB (billed on a pay-per-use basis from April 10, 2023, 23:59:59 to April 20, 2023, 23:59:59)
  - Public network bandwidth: 6 Mbit/s
  - Billed duration: one month

NOTICE

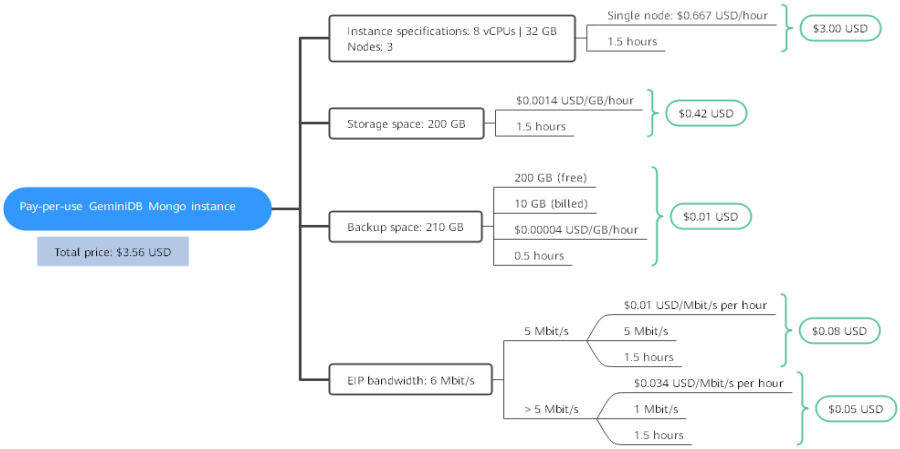
Unit prices in this example are used for reference only. The prices shown here are only estimates. As unit prices change from time to time, the prices shown here will differ from actual prices. For details, see the data released on the Huawei Cloud official website.

Pay-per-use

From March 18, 2023, 15:30:00 to March 20, 2023, 09:00:00, a GeminiDB Mongo instance with specifications 4 vCPUs | 16 GB was used for 41.5 hours, so the price would be calculated as follows.

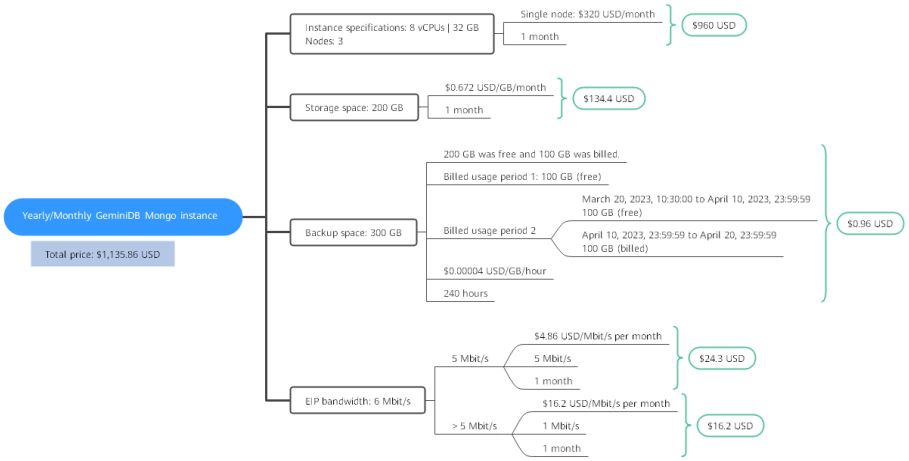


From March 20, 2023, 09:00:00 to March 20, 2023, 10:30:00, a GeminiDB Mongo instance with specifications 8 vCPUs | 32 GB was used for 1.5 hours, so the price would be calculated as follows.



Yearly/Monthly

From March 20, 2023, 10:30:00 to April 20, 2023, 23:59:59, a GeminiDB Mongo instance purchased using yearly/monthly billing was used for one month, so the price would be calculated as follows.



From March to April, the total price of this GeminiDB Mongo instance is $1,190.18 USD (50.76 + 3.56 + 1135.86).

2.5 Billing Mode Changes

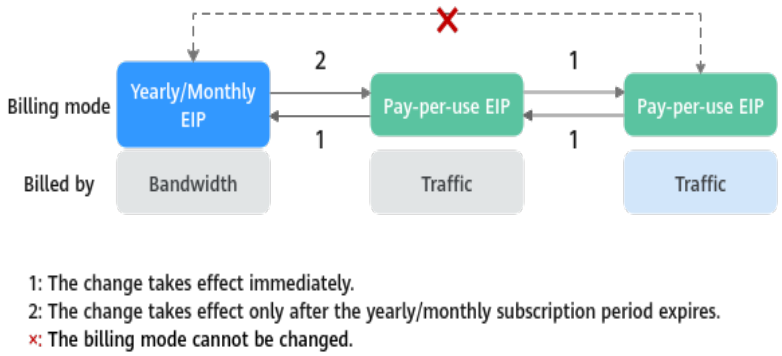
2.5.1 Overview

After purchasing a GeminiDB Mongo instance, you can change the billing mode if it no longer meets your needs. **Table 2-7** lists changeable billing items of the GeminiDB Mongo instance.

Table 2-7 Changeable billing items of GeminiDB Mongo instances

Billing Item	Change Description	Reference
Instance specifications (vCPUs and nodes)	Changing the billing mode of a GeminiDB Mongo instance includes the changes to compute resources (vCPUs and nodes). • Change from pay-per-use to yearly/monthly to enjoy lower prices. • Change from yearly/monthly to pay-per-use to use the GeminiDB Mongo instance more flexibly. **NOTE** Such a change takes effect only after the yearly/monthly subscription ends.	• **Pay-per-Use to Yearly/Monthly** • **Yearly/Monthly to Pay-per-Use**
EIP	• A yearly/monthly EIP can be changed to a pay-per-use EIP billed by bandwidth after the yearly/monthly subscription ends. • A pay-per-use EIP billed by bandwidth can be changed to a yearly/monthly EIP. • Pay-per-use EIPs billed by bandwidth can be changed to pay-per-use EIPs billed by traffic, and pay-per-use EIPs billed by traffic can be changed to pay-per-use EIPs billed by bandwidth. For details, see **Figure 2-8**.	• **Pay-per-Use to Yearly/Monthly** • **Yearly/Monthly to Pay-per-Use**

Figure 2-8 EIP billing mode change



2.5.2 Pay-per-Use to Yearly/Monthly

If you have a pay-per-use GeminiDB Mongo instance that you expect to use for a long time, you can change it to yearly/monthly billing to reduce costs. Doing so will create an order. After you pay for the order, yearly/monthly billing will be applied immediately.

Suppose you bought a pay-per-use GeminiDB Mongo instance at 15:29:16 on April 18, 2023 and changed it to yearly/monthly billing at 16:30:30 on the same day. After you paid for the order, yearly/monthly billing was applied immediately. On the **Billing Center** > **Billing** page, three line items were generated.

- Pay-per-use expenditures for 15:29:16 to 16:00:00 on April 18, 2023
- Pay-per-use expenditures for 16:00:00 to 16:30:30 on April 18, 2023
- Yearly//monthly expenditure generated on April 18, 2023, 16:30:30

Constraints

Resources such as EIPs that are used by an instance may not support the change with this instance. For details about their billing mode change rules and handling methods, see **Table 2-8**.

Table 2-8 EIP billing mode change rules

Resource	Billing Mode	Billed By	Bandwidth Type	Change to Yearly/ Monthly Billing with GeminiDB Mongo Instance	Handling Measure
EIP	Pay-per-use	Bandwidth	Dedicated	Supported	Change the EIP to yearly/ monthly billing on the EIP console. For details, see **Changing EIP Billing Mode**.
EIP	Pay-per-use	Traffic	Dedicated	Not supported	An EIP that is billed by traffic on a pay-per-use basis cannot be directly changed to be billed on a yearly/monthly basis. To change this: 1. Change the EIP to be billed by bandwidth on a pay-per-use basis. 2. Change the EIP to be billed on a yearly/ monthly basis. For details, see **Changing EIP Billing Mode**.

Prerequisites

- The billing mode of the instance is pay-per-use.
- The instance status is **Available**.

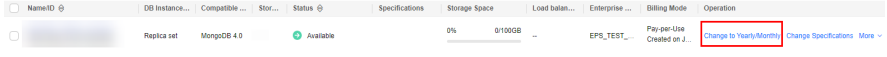
Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

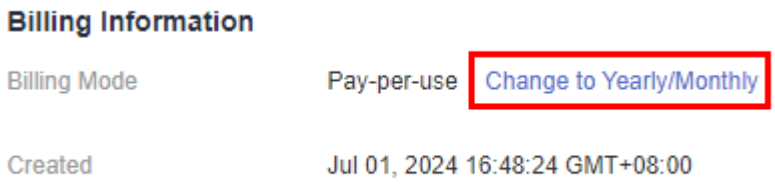
Step 3 On the **Instances** page, locate the instance whose billing mode you want to change and click **Change to Yearly/Monthly** in the **Operation** column.

Figure 2-9 Changing pay-per-use to yearly/monthly



Alternatively, click the instance name to go to the **Basic Information** page. In the **Billing Information** area, click **Change to Yearly/Monthly** in the **Billing Mode** field.

Figure 2-10 Changing from pay-per-use to yearly/monthly



NOTE

The billing mode of multiple instances can be changed in batches. Perform the following steps:

1. Select the instances whose billing mode you want to change.
2. Click **Change to Yearly/Monthly** above the instance list.

Step 4 On the displayed page, specify a subscription duration in month. The minimum duration is one month.

If you do not need to modify your settings, click **Pay** to go to the payment page.

Step 5 Select a payment method and click **Confirm**.

Step 6 View results on the **Instances** page.

In the upper right corner of the instance list, click to refresh the list. The instance status will become **Available** after the change is successful. The billing mode changes to **Yearly/Monthly**.

----End

2.5.3 Yearly/Monthly to Pay-per-Use

After creating a yearly/monthly GeminiDB Mongo instance, you can change it to pay-per-use for more flexibility, and you can recoup part of what you paid for the subscription.

Suppose you bought a yearly/monthly GeminiDB Mongo instance at 15:29:16 on April 18, 2023 and changed it to pay-per-use billing at 16:30:00 on May 18, 2023. On the **Billing Center** > **Billing** page, bills information is generated as follows:

- Yearly/Monthly expenditures for 15:29:16 on April 18 to 23:59:59 on May 18, 2023
- Pay-per-use expenditures for 23:59:59 on May 18, 2023 to the end time of pay-per-use billing. A bill was generated every hour.

NOTE

The pay-per-use billing mode will take effect only after the yearly/monthly subscription has expired. Auto-renewal will not be in effect.

Constraints

Resources such as EIPs that are used by an instance may not support the change with this instance. For details about their billing mode change rules and handling methods, see **Table 2-9**.

Table 2-9 EIP billing mode change rules

Resource	Billing Mode	Billed By	Bandwidth Type	Change to Pay-per-Use Billing with GeminiDB Mongo Instance	Handling Measure
EIP	Yearly/Monthly	Bandwidth	Dedicated	Not supported	Change the EIP to yearly/monthly billing on the EIP console. For details, see **Changing EIP Billing Mode**.
EIP	Yearly/Monthly	Traffic	Dedicated	Not supported	An EIP billed on a yearly/monthly basis cannot be directly changed to be billed by traffic on a pay-per-use basis. To change this: 1. Change the EIP to be billed by bandwidth on a pay-per-use basis. 2. Change the EIP to be billed by traffic on a pay-per-use basis. For details, see **Changing EIP Billing Mode**.

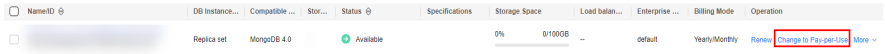
Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose billing mode you want to change and click **More** > **Change to Pay-per-Use** in the **Operation** column.

Figure 2-11 Changing from yearly/monthly to pay-per-use



NOTE

The billing mode of multiple pay-per-use instances can be changed in batches. Perform the following steps:

1. Select the instances whose billing mode you want to change.
2. Click **More** > **Change to Pay-per-Use** in the **Operation** column

Step 4 On the displayed page, confirm the instance information and click **Change to Pay-per-Use**. The billing mode will change to pay-per-use after the instance expires. Auto renewal will be disabled after the billing mode of your instances change to pay-per-use. Exercise caution when performing this operation.

Step 5 After you submit the change, check whether a message is displayed in the **Billing Mode** column, indicating that the billing mode will be changed to pay-per-use after the subscription expires.

Step 6 To cancel the change, choose **Billing** > **Renewal** to enter the Billing Center. On the **Renewals** page, locate the instance and click **More** > **Cancel Change to Pay-per-Use**.

Step 7 In the displayed dialog box, click **Yes**.

----End

2.6 Renewing Subscriptions

2.6.1 Overview

When to Renew Subscriptions

If a yearly/monthly instance is about to expire but you want to continue using it, you need to renew the instance subscription within a specified period, or resources, such as vCPUs and memory, will be automatically released, and data will be lost and cannot be restored.

Only yearly/monthly instance subscriptions can be renewed. If you use pay-per-use instances, just ensure that your account has a valid payment method configured or a top-up account with a sufficient balance.

If you renew the instance before it expires, resources will be retained and you can continue using the instance. For details about statuses after instances have expired and the associated impacts, see **Impact of Expiration**.

How to Renew Subscriptions

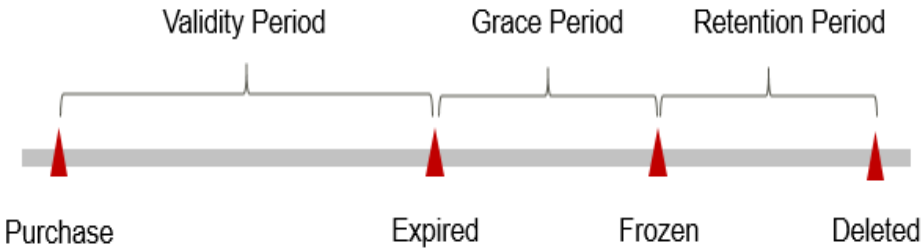
You can renew a yearly/monthly instance manually or automatically.

Table 2-10 Renewing a yearly/monthly instance

Method	Description
Manually Renewing an Instance	You can renew a yearly/monthly instance anytime on the console before it is automatically deleted.
Auto-renewing an Instance	You can enable auto-renewal to automatically renew the instance before it expires. This prevents resources from being deleted in case you forget to renew a subscription.

You can select a method to renew a yearly/monthly instance based on the phase the instance is currently in.

Figure 2-12 Selecting a renewal method based on the instance's current phase



2.6.2 Manually Renewing an Instance

You can renew a yearly/monthly instance anytime on the console before it is automatically deleted.

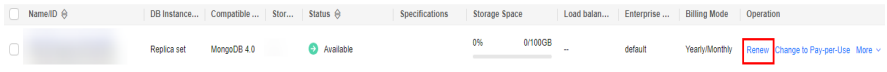
Renewing an Instance on the Console

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

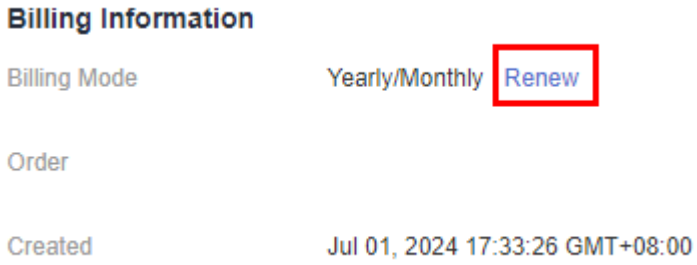
Step 3 On the **Instances** page, locate the instance that you want to renew and click **Renew** in the **Operation** column.

Figure 2-13 Renewal button



Alternatively, click the instance name to go to the **Basic Information** page. In the **Billing Information** area, click **Renew** next to the **Billing Mode** field.

Figure 2-14 Renewal button



NOTE

To renew multiple yearly/monthly instances at a time, perform the following steps:

1. Select the yearly/monthly instances to be renewed.
2. Click **Renew** above the instance list.

Step 4 On the displayed page, renew the instances.

----End

Renewing a Subscription in Billing Center

Step 1 Log in to the management console.

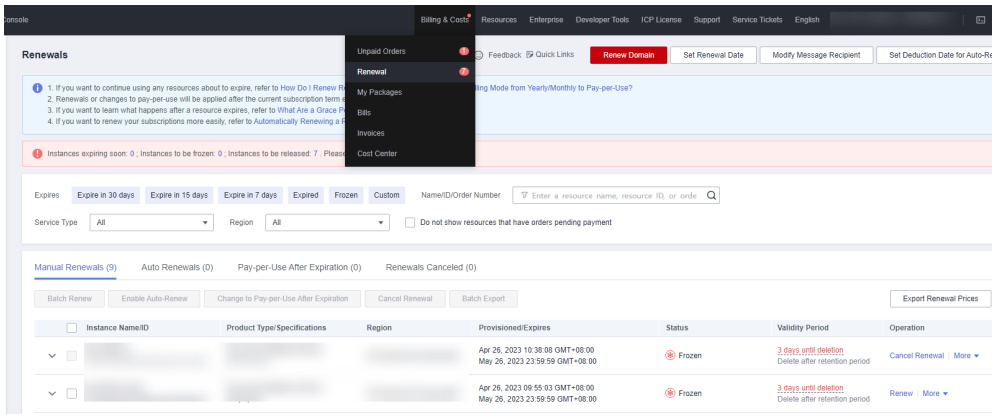
Step 2 Hover over **Billing & Costs** in the upper part of the console and choose **Renewal** from the drop-down list.

The **Renewals** page is displayed.

Step 3 Select the search criteria.

On the **Manual Renewals**, **Auto Renewals**, **Pay-per-Use After Expiration**, and **Renewals Canceled** pages, you can view the instances to be renewed.

Figure 2-15 Renewal management

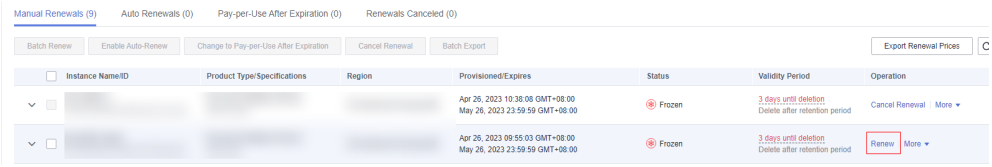


You can move all resources that need to be manually renewed to the **Manual Renewals** tab page. For details, see **Restoring to Manual Renewal**.

Step 4 Manually renew resources.

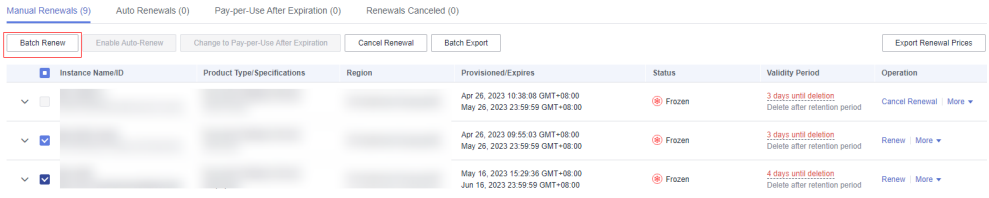
- Individual renewal: Locate an instance that you want to renew and click **Renew** in the **Operation** column.

  Figure 2-16 Individual renewal



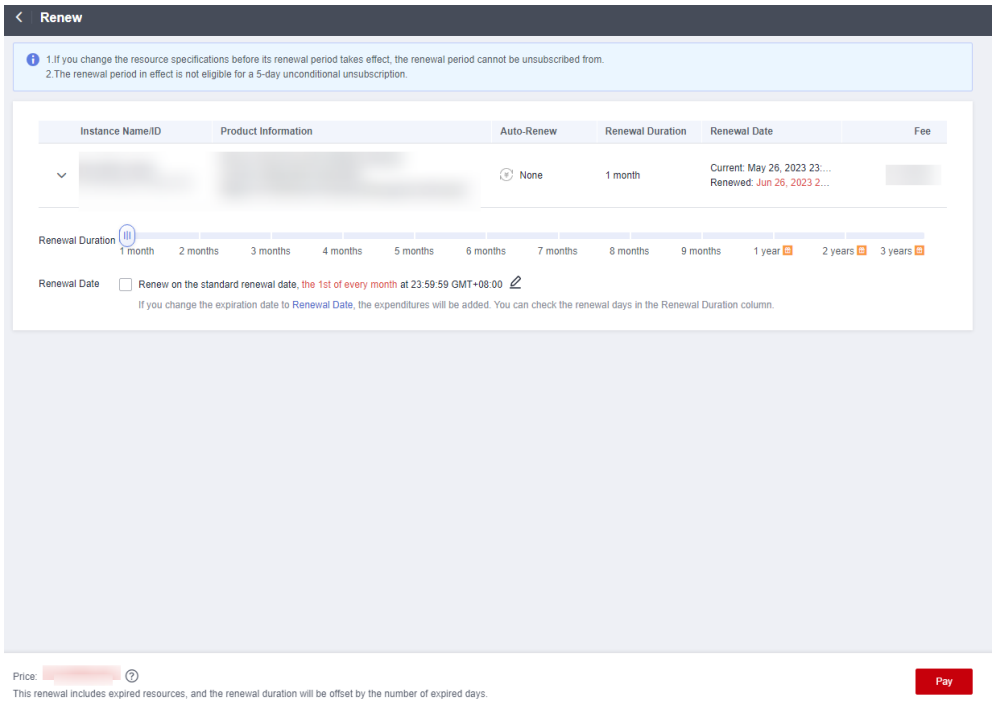
- Batch renewal: Select multiple instances that you want to renew and click **Batch Renew** in the upper left corner.

  Figure 2-17 Batch renewal



Step 5 Select a renewal duration and optionally select **Renew on the standard renewal date**. For details, see **Setting the Same Renewal Day for Yearly/Monthly Resources**. Confirm the price and click **Pay**.

Figure 2-18 Confirming renewal



Step 6 Select a payment method and make your payment. Once the order is paid for, the renewal is complete.

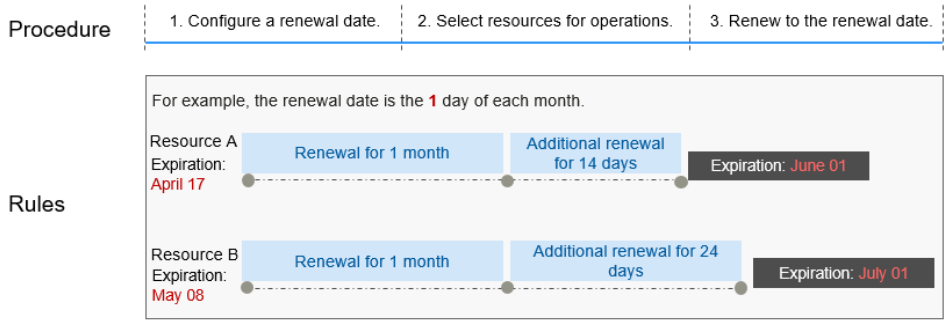
----End

Setting the Same Renewal Day for Yearly/Monthly Resources

If the instances have different expiry dates, you can set the same renewal day, for example, the first day of each month, to make it easier to manage renewals.

In **Figure 2-19**, a user sets the same renewal day for two resources that will expire at different dates.

Figure 2-19 Setting the same renewal day for resources with different expiry dates



For more details, see **Setting a Renewal Date**.

2.6.3 Auto-renewing an Instance

Auto-renewal can prevent instances from being automatically deleted if you forget to manually renew them. The auto-renewal rules are as follows:

- The first auto-renewal date is based on when an instance expires and the billing cycle.
- The auto-renewal period of an instance depends on the subscription term.
  - Monthly subscriptions renew each month.
  - Yearly subscriptions renew each year.
- You can enable auto-renewal any time before an instance expires. By default, the system will make the first attempt to charge your account for the renewal at 03:00 seven days before the expiry date. If this attempt fails, it will make another attempt at 03:00 every day until the subscription is renewed or expired.
- After auto-renewal is enabled, you can still renew the instance manually if you want to. After a manual renewal is complete, auto-renewal is still valid, and the renewal fee will be deducted from your account seven days before the new expiry date.
- By default, the renewal fee is deducted from your account seven days before the new expiry date. You can change this auto-renewal payment date as required.

For more information about auto-renewal rules, see **Auto-Renewal Rules**.

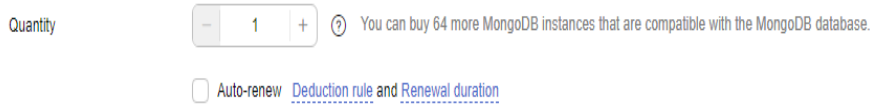
Prerequisites

Your yearly/monthly instance is not expired.

Enabling Auto-Renewal During Purchase

You can enable auto-renewal on the instance purchase page, as shown in **Figure 2-20**. For details, see **Buying an Instance**.

Figure 2-20 Enabling auto-renewal



Enabling Auto-Renewal on the Renewals Page

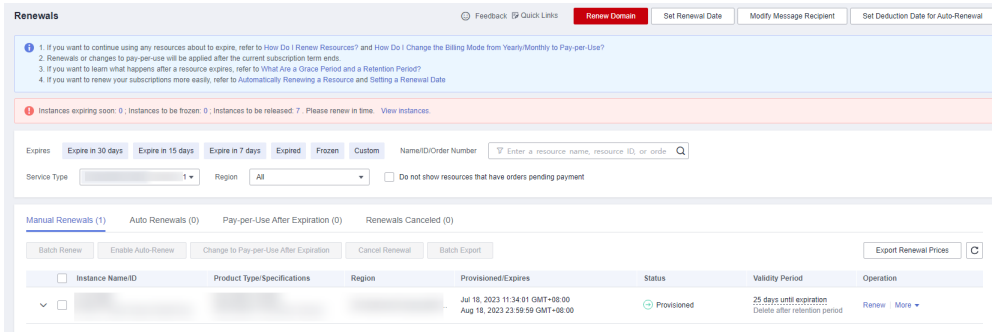
Step 1 Log in to the management console.

Step 2 Hover over **Billing & Costs** in the upper part of the console and choose **Renewal** from the drop-down list.

Step 3 Select the search criteria.

- On the **Auto Renewals** page, you can view the resources that auto-renewal has been enabled for.
- You can enable auto-renewal for resources on the **Manual Renewals**, **Pay-per-Use After Expiration**, and **Renewals Canceled** pages.

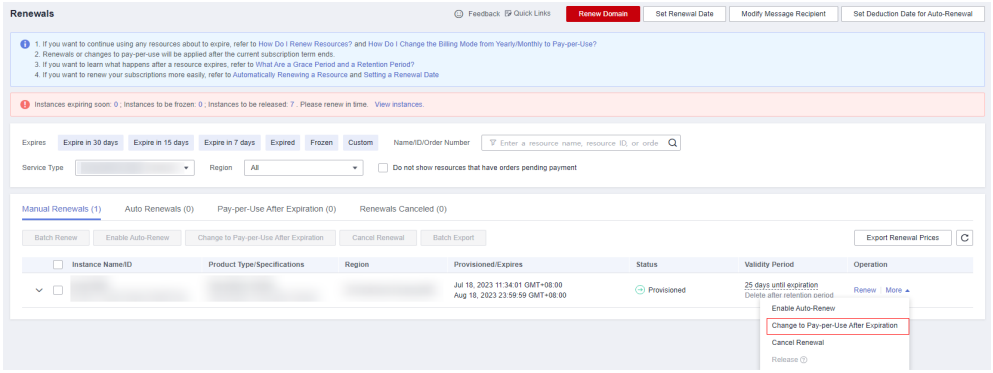
Figure 2-21 Renewal management



Step 4 Enable auto-renewal for yearly/monthly resources.

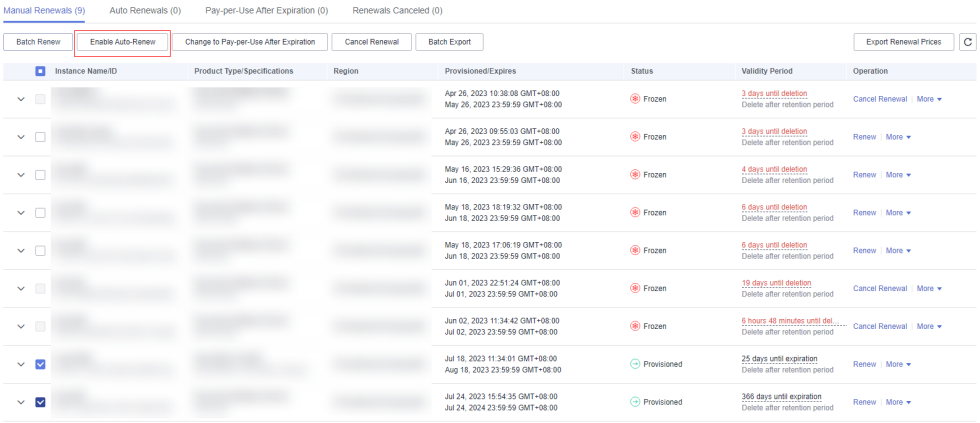
- Enabling auto-renewal for a single instance: Locate the instance that you want to enable auto-renewal for and choose **More** > **Enable Auto-Renew** in the **Operation** column.

Figure 2-22 Enabling auto-renewal for an instance



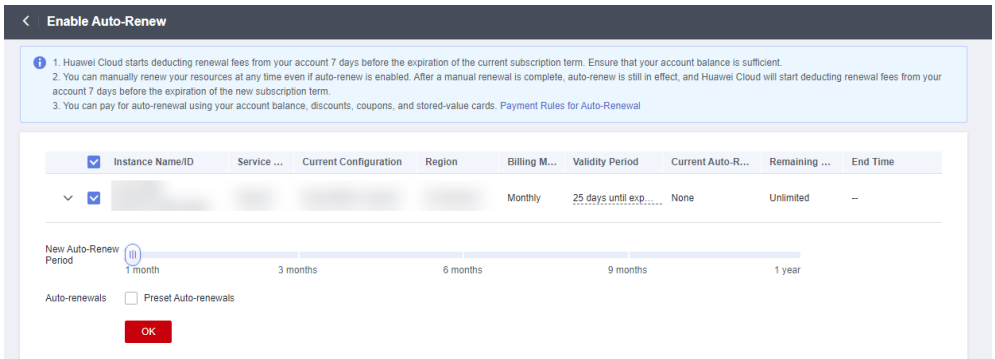
- Enabling auto-renewal for multiple instances at a time: Select the instances that you want to enable auto-renewal for and click **Enable Auto-Renew** above the list.

Figure 2-23 Enabling auto-renewal for multiple instances



Step 5 Select a renewal period, specify the auto-renewal times, and click **Pay**.

Figure 2-24 Enabling auto-renewal



----End

2.7 Bills

You can view the resource usage and bills for different billing cycles on the **Bills** page in the Billing Center.

Bill Generation

Transaction records for yearly/monthly subscriptions are generated immediately after being paid for.

A pay-per-use resource is billed by the hour, day, or month, depending on the resource's usage type. The GeminiDB Mongo instance usage is billed by the hour. For details, see **Bill Run for Pay-per-Use Resources**

You are not charged immediately after a record is generated. For example, if a pay-per-use GeminiDB Mongo instance (which is billed on an hourly basis) is deleted at 08:30, you will still have expenditures for the 08:00 to 09:00 hour. However, you will not likely be billed for the 08:00 to 09:00 hour until about 10:00. On the **Bills** page of the Billing Center, select the **Bill Details** tab. **Expenditure Time** in the bill indicates the time when the pay-per-use resource is used.

Viewing Bills of a Specific Resource

[Method 1: Use the instance ID to search for a bill.]

Step 1 Log in to the management console and click **Databases** > **GeminiDB Mongo API**.

Step 2 On the **Instances** page, locate the instance whose bill you want to view and click its name.

Step 3 Click the icon shown in the figure below to copy the instance ID.

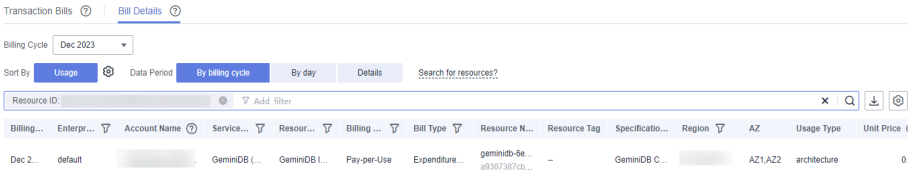
Figure 2-25 Copying the instance ID

Step 4 On the top menu bar, choose **Billing & Costs** > **Bills**.

The **Bills** page is displayed.

Step 5 Choose **Transactions and Detailed Bills** > **Bill Details**. On the displayed page, select **Resource ID** as the filter criteria, enter the obtained instance ID, and click the icon.

Figure 2-26 Searching for a bill



By default, the bill details are displayed by usage and billing cycle. You can choose other display options as required. For details, see **Bill Details**.

----End

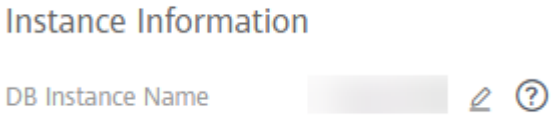
[Method 2: Use the resource name to search for a bill.]

Step 1 Log in to the management console and click **Databases** > **GeminiDB Mongo API**.

Step 2 On the **Instances** page, locate the instance whose bill you want to view and click its name.

Step 3 On the **Basic Information** > **Instance Information** page, obtain the instance name.

Figure 2-27 Copying the instance name

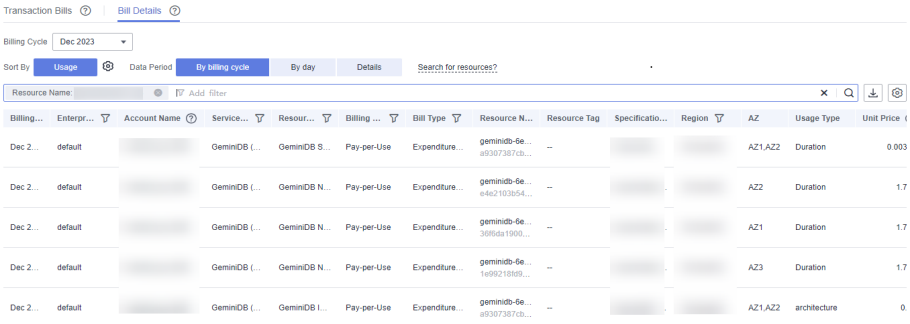


Step 4 On the top menu bar, choose **Billing & Costs** > **Bills**.

The **Bills** page is displayed.

Step 5 Choose **Transactions and Detailed Bills** > **Bill Details**. On the displayed page, select **Resource Name** as the filter criteria, enter the obtained instance ID, and click the icon.

Figure 2-28 Searching for a bill



By default, the bill details are displayed by usage and billing cycle. You can choose other display options as required. For details, see **Bill Details**.

----End

Scenario Example: Checking the Consistency of the Actual Usage and Billed Usage

Assume that you purchased a pay-per-use GeminiDB Mongo instance at 10:09:06 on April 8, 2023 and deleted it later that day, at 12:09:06.

- Transaction Records

  Pay-per-use GeminiDB Mongo instance usage is calculated by the second and but billed on an hourly basis. You can check the transaction records against the actual usage. The billed resources are billed separately. For details, see **Table 2-11**.

  Table 2-11 GeminiDB Mongo transaction records

Service Type	GeminiDB Mongo
Resource Type	GeminiDB Mongo storage
Billing Mode	Pay-per-use
Expenditure Time	For the period of time from 10:09:06 to 12:09:06 on April 08, 2023, 6 transaction records would be generated for the resource usage in the following periods: • 2023/04/08 10:09:06 - 2023/04/08 11:00:00 • 2023/04/08 11:00:00 - 2023/04/08 12:00:00 • 2023/04/08 12:00:00 - 2023/04/08 12:09:06
List Price	List price on the official website = Usage x Unit price x Capacity The GeminiDB Mongo instance was used for 3,054 seconds in the first period, and the unit price can be obtained on the **Pricing Details** page. The list price for the first period = (3054 ÷ 3600) × 0.0014 × 40 = $0.04750667 USD. Similarly, you can calculate the instance list price for the other periods.
Discounted Amount	Discounts offered for cloud services, for example, commercial discounts, partner authorized discounts, and promotional discounts. It is the discounted amount based on the list price.
Truncated Amount	Billing of Huawei Cloud is calculated to the 8th decimal place. However, the amount due is truncated to the 2nd decimal place. The third and later decimal places are referred to as the truncated amounts. Take the first period as an example. The truncated amount is $0.00750667 USD.
Amount Due	Amount due = List price – Discount amount – Truncated amount Take the first period as an example. If the discount amount is 0, the amount due is $0.04 USD (0.04750667 - 0 - 0.00750667).

- Bill details of the GeminiDB Mongo instance

  Bill details can display in multiple ways. By default, the bill details of a resource are displayed by usage and by billing cycle. **Table 2-12** illustrates the GeminiDB Mongo instance bill details, which can be used to check against the actual usage.

  Table 2-12 Bill details of the GeminiDB Mongo instance

Service Type	GeminiDB Mongo
Resource Type	GeminiDB Mongo storage
Billing Mode	Pay-per-use
Resource Name/ID	Name and ID of a specific GeminiDB Mongo instance Example: nosql-b388 and 21e8811a64bf4de88bc2e2556da17983in12
Specifications	GeminiDB Mongo storage
Usage Type	Duration for a pay-per-use GeminiDB Mongo instance
Unit Price	When pay-per-use billing is used, the unit price is only provided if the amount is equal to the usage multiplied by the unit price. No unit price is provided in other pricing modes, for example, tiered pricing. You can search for the unit price for pay-per-use GeminiDB Mongo instances on **Product Pricing Details**.
Unit	Displayed on the **Product Pricing Details** page. Example: USD/GB/hour.
Usage	Depends on the unit of the unit price, which is USD/GB/hour. Storage usage is billed by the hour. Example: 2 hours.
Usage Unit	Hour
List Price	List price on the official website = Usage x Unit price x Capacity The GeminiDB Mongo instance is used for 2 hours in total, and the unit price is obtained on the **Product Pricing Details** page. The list price = 2 * 0.0014 * 40 = $0.112 USD.
Discounted Amount	Discounts offered for cloud services, for example, commercial discounts, partner authorized discounts, and promotional discounts. It is the discounted amount based on the list price.

Amount Due	Amount that should be paid for used cloud services after discounts are applied.

2.8 Arrears

If your configured payment method is unable to pay for your bill, your account will be in arrears. You will need to update you payment method or to top up your account in a timely manner if you want to continue using your instance resources.

Arrears Reason

If you do not have yearly/monthly instances, your account falls into arrears any time your configured payment method is unable to pay for the used resources on the pay-per-use basis.

Arrears Impact

- Yearly/Monthly

  This is a pre-paid billing mode, so you can continue using yearly/monthly GeminiDB Mongo resources even if your account is in arrears. However, you cannot perform operations such as purchasing GeminiDB Mongo instances, upgrading instance specifications, and renewing subscriptions, because they will generate new expenditures.

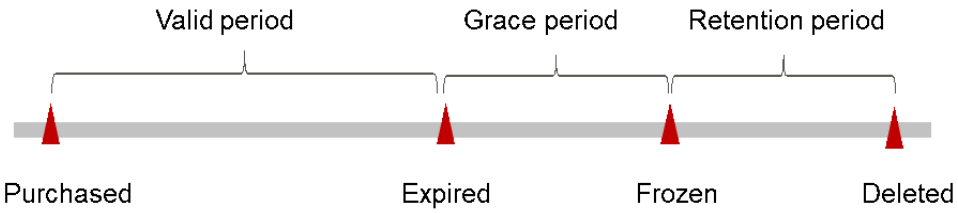
- Pay-per-Use

  If your configured payment method is unable to pay a bill for pay-per-use resources, the resources enter a grace period. After you top up your account, Huawei Cloud will bill you for expenditures generated by the resources during the grace period. You can view the expenditures on the **Overview** page of the Billing Center.

  If your account is still in arrears after the grace period ends, the resources enter the retention period and their status turns to **Frozen**. You cannot perform any operations on these resources.

  After the retention period ends, the compute resources (vCPUs and memory) and EIPs will be released and cannot be restored.

Figure 2-29 Lifecycle of a pay-per-use instance



NOTE

The grace period and retention period are both 15 days.

Avoiding and Handling Arrears

Make sure you have a valid payment method configured as soon as possible after your account is in arrears. For details, see **Topping Up an Account**.

If a GeminiDB Mongo instance is no longer used, you can delete it to avoid generating further expenditures.

To help make sure your account never falls into arrears, you can configure the **Balance Alert** on the **Overview** page of the Billing Center. Then, any time an expenditure quota drops to below the threshold you specify, Huawei Cloud automatically notifies you by SMS or email.

2.9 Billing Termination

Yearly/Monthly Resources

When you purchase a yearly/monthly resource, such as a yearly/monthly GeminiDB Mongo instance, you make a one-time up-front payment. By default, the billing automatically stops when the purchased subscription expires.

- If a yearly/monthly resource is no longer needed before the subscription expires, you can unsubscribe from the resource. The system will return a certain amount of money to your account based on whether the resource is subject to five-day unconditional unsubscription or whether cash coupons or discount coupons are used. For details about unsubscription rules, see **Unsubscriptions**.
- If you have enabled auto-renewal but no longer wish to automatically renew the subscription, disable it before the auto-renewal date (7 days before the expiration date by default) to avoid unexpected expenditures.

Pay-per-Use Resources

If pay-per-use resources, such as pay-per-use GeminiDB Mongo instances, are no longer required, delete them in a timely manner.

Searching for Resources from Bills and Stopping Billing

To ensure that all related resources are deleted, you can search the billing records by resource ID, and then delete the resources you identify in this way.

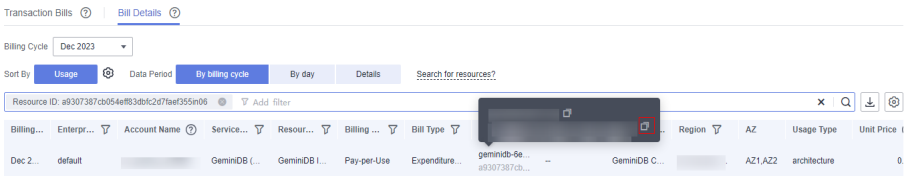
[Method 1: Use the resource ID in the bill to search for the resource.]

Step 1 Log in to the management console. On the top menu bar, choose **Billing & Costs** > **Bills**.

The **Bills** page is displayed.

Step 2 Choose **Transactions and Detailed Bills** > **Bill Details**, and click the icon shown in the following figure to copy the resource ID.

Figure 2-30 Copying the resource ID



Step 3 Log in to the management console and click **Databases** > **GeminiDB Mongo API**.

Step 4 Select the region where the resource is located, select **Instance ID** and enter the resource ID copied in **Step 2**, and click the icon to search for the resource.

Figure 2-31 Searching for resources

Step 5 Locate the instance you want to delete and click **More** > **Delete** in the **Operation** column. Ensure that the resource is not found in the list.

NOTE

You are billed one hour after the resource usage is calculated, so a bill may still be generated after the pay-per-use resource is deleted. For example, if you delete an instance (which is billed on an hourly basis) at 08:30, the expenditures for that hour from 08:00 to 09:00 are usually not billed until about 10:00.

----End

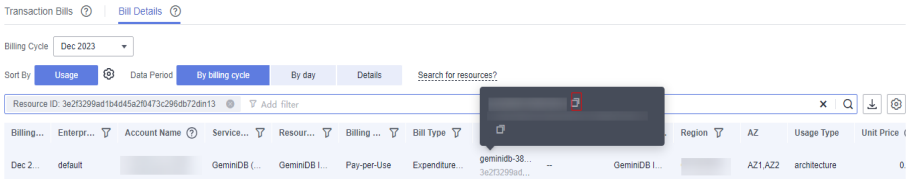
[Method 2: Use the resource name in the bill to search for the resource.]

Step 1 Log in to the management console. On the top menu bar, choose **Billing & Costs** > **Bills**.

The **Bills** page is displayed.

Step 2 Choose **Transactions and Detailed Bills** > **Bill Details**, and click the icon shown in the following figure to copy the resource name.

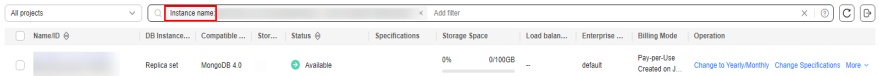
Figure 2-32 Copying the resource name



Step 3 Log in to the management console and click **Databases** > **GeminiDB Mongo API**.

Step 4 Enter the instance name copied in **Step 2** in the search box and click .

Figure 2-33 Searching for resources



Step 5 Locate the instance you want to delete and click **More** > **Delete** in the **Operation** column. Ensure that the resource is not found in the list.

NOTE

You are billed one hour after the resource usage is calculated, so a bill may still be generated after the pay-per-use resource is deleted. For example, if you delete an instance (which is billed on an hourly basis) at 08:30, the expenditures for that hour from 08:00 to 09:00 are usually not billed until about 10:00.

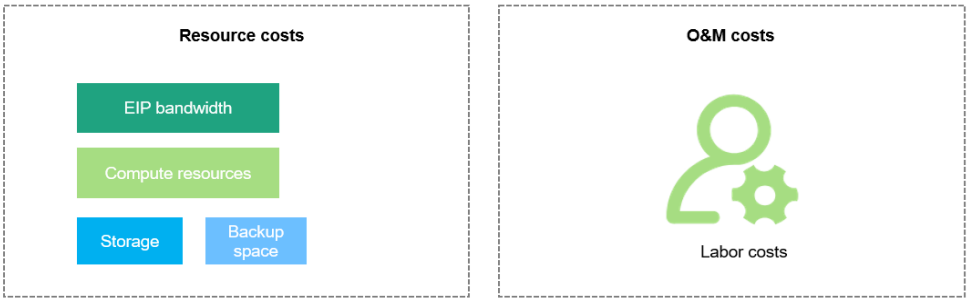
----End

2.10 Cost Management

2.10.1 Cost Composition

GeminiDB Mongo costs consist of two parts:

- Resource costs: costs of compute and storage resources. For details, see **Billing Modes**.
- O&M costs: labor costs incurred during the use of GeminiDB Mongo.



2.10.2 Cost Allocation

A good cost accountability system is a prerequisite for cost management. It ensures that departments, business teams, and owners are accountable for their respective cloud costs. An enterprise can allocate cloud costs to different teams or projects so as to have a clear picture of their respective costs.

Huawei Cloud **Cost Center** provides various tools for you to group costs in different ways. You can experiment with these tools and find a way that works best for you.

- **By linked account**

  The enterprise master account can manage costs by grouping the costs of its member accounts by linked account. For details, see **Viewing Costs by Linked Account**.

- **By enterprise project**

  Before allocating costs, enable Enterprise Project Management Service (EPS) and plan your enterprise projects based on your organizational structure or service needs. When purchasing cloud resources, select an enterprise project

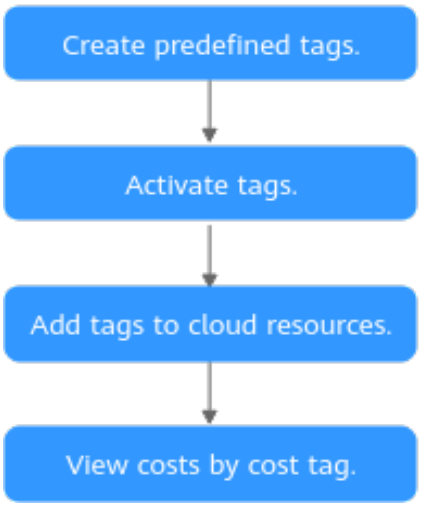
so that the costs of resources will be allocated to the selected enterprise project. For details, see **Viewing Costs by Enterprise Project**.

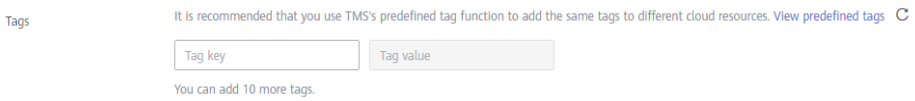
Figure 2-34 Selecting an enterprise project

- **By cost tag**

  You use tags to sort your Huawei Cloud resources in a variety of different ways, for example, by purpose, owner, or environment. The following is the process of managing costs by predefined tags (recommended).

  Figure 2-35 Adding a tag

  For details, see **Viewing Costs by Cost Tag**.

- **By cost category**

  You can use cost categories provided by **Cost Center** to split shared costs. Shared costs are the costs of resources (compute, network, storage, or resource packages) shared across multiple departments or the costs that cannot be directly split by cost tag or enterprise project. These costs are not directly attributable to a singular owner, and they cannot be categorized into a singular cost type. In this case, you can define cost splitting rules to fairly allocate these costs among teams or business units. For details, see **Viewing Cost By Cost Category**.

2.10.3 Cost Analysis

To precisely control and optimize your costs, you need a clear understanding of what parts of your enterprise incurred different costs. **Cost Center** visualizes your original costs and amortized costs using various dimensions and display filters for

cost analysis so that you can analyze the trends and drivers of your service usage and costs from a variety of perspectives or within different defined scopes.

You can also use cost anomaly detection provided by **Cost Center** to detect unexpected expenses in a timely manner. In this way, costs can be monitored, analyzed, and traced.

For details, see **Performing Cost Analysis to Explore Costs and Usage** and **Enabling Cost Anomaly Detection to Identify Anomalies**.

2.10.4 Cost Optimization

You can identify resources with high costs based on the analysis results in the cost center, determine the causes of high costs, and take optimization measures accordingly.

Resource rightsizing

- View GeminiDB Mongo monitoring metrics on Cloud Eye, such as the CPU, memory, and disk usage. If the current configuration is too high, you can reduce the configuration by changing specifications.
- Monitor idle GeminiDB Mongo resources and delete idle instances in a timely manner.

Billing mode selection

Different types of services have different requirements on resource usage periods, so the most economical billing mode for one resource may not be the best option for another resource.

- For mature services that tend to be stable for the long term, select yearly/monthly billing.
- For short-term, unpredictable services that experience traffic bursts and cannot afford to be interrupted, select pay-per-use billing.
- Monitor the lifecycle of instances and renew yearly/monthly resources that are about to expire in a timely manner.

2.11 Billing FAQs

2.11.1 What Are the Differences Between Yearly/Monthly and Pay-per-Use Billing?

Yearly/Monthly is a prepaid billing mode in which resources are billed based on the service duration. This cost-effective mode is ideal when the duration of resource usage is predictable. It is recommended for long-term users.

Pay-per-use billing is a postpaid payment mode. This billing mode allows you to make or cancel subscriptions at any time. Pricing is listed on a per-hour basis, but bills are calculated based on the actual usage duration.

2.11.2 Can I Switch Between Yearly/Monthly and Pay-per-Use Billing?

You can change the billing mode of your instance from yearly/monthly to pay-per-use or vice versa.

- For details about how to change the billing mode from yearly/monthly to a pay-per-use, see **Yearly/Monthly to Pay-per-Use**.
- For details about how to change the billing mode from pay-per-use to yearly/monthly, see **Pay-per-Use to Yearly/Monthly**.

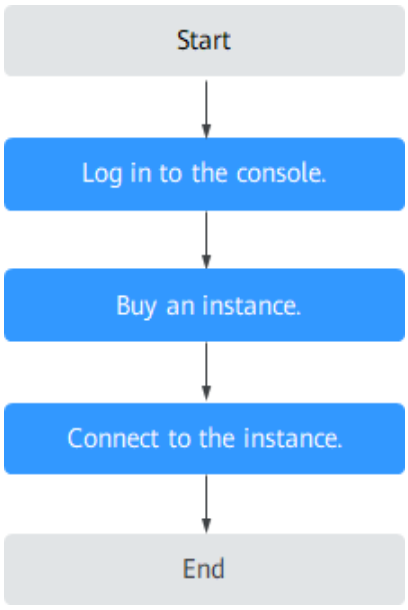
3 Getting Started with GeminiDB Mongo API

3.1 Service Overview

This section describes how to buy an instance and then connect to and manage it.

Process

Figure 3-1 Flowchart



Procedure

The process of buying and using an instance involves the following steps:

Step 1: Log in to the GeminiDB Mongo API console.

Step 2: **Buy an instance.**

Step 3: **Connect to the instance.**

3.2 Buying a Replica Set Instance

This section describes how to buy a replica set instance that is compatible with MongoDB APIs. You can customize your replica set instance with appropriate computing capability and storage space based on service requirements.

Prerequisites

- You have registered a Huawei Cloud account.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click **Buy DB Instance**.

Step 4 On the displayed page, select a billing mode, specify instance specifications and click **Next**.

Figure 3-2 Billing mode and basic information

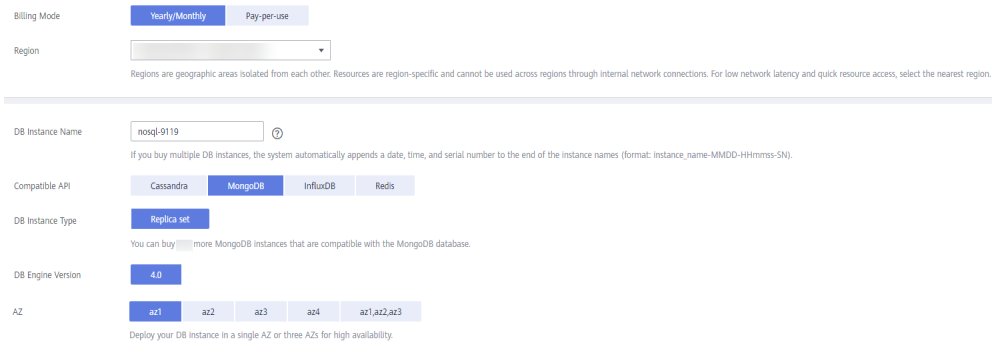


Table 3-1 Billing mode

Parameter	Description
Billing Mode	Method that the instance is billed in. The value can be **Yearly/Monthly** or **Pay-per-use**. • Yearly/Monthly – In this mode, specify **Required Duration** at the bottom of the page. The system deducts the fees incurred from your account based on the service price. – If you do not need such an instance any longer after it expires, change the billing mode to pay-per-use to optimize costs. For details, see **Changing the Billing Mode from Yearly/Monthly to Pay-per-Use**. **NOTE** Yearly/Monthly instances cannot be deleted directly. If such an instance is no longer required, unsubscribe from it. For details, see **Unsubscribing from a Yearly/Monthly DB Instance**. • Pay-per-use – If you select this billing mode, you are billed based on how much time the instance is in use. – If you expect to use an instance for a long period of time, change its billing mode to yearly/monthly to optimize costs. For details, see **Changing the Billing Mode from Pay-per-Use to Yearly/Monthly**.

Table 3-2 Basic information

Parameter	Description
Region	The region where the instance is deployed. **NOTICE** Select the region nearest where you will be accessing the DB from so latency can be kept to a minimum and response time will be faster. Instances deployed in different regions cannot communicate with each other through a private network. After you buy an instance, you cannot change its region.
DB Instance Name	The instance name: • Can be the same as an existing instance name. • Can include 4 to 64 bytes and must start with a letter. It is case-sensitive and allows only letters, digits, hyphens (-), and underscores (_). After an instance is created, you can change its name. For details, see **Modifying the Name of an Instance**.
Compatible API	MongoDB
DB Instance Type	Replica set
DB Engine Version	4.0

Parameter	Description
AZ	Availability zone where the instance is created. An AZ is a part of a region with its own independent power supplies and networks. AZs are physically isolated but can communicate through an internal network. An instance can be deployed in one or three AZs. • If you want to deploy an instance in a single AZ, select one AZ. • If you want to deploy an instance across AZs for disaster recovery, select three AZs. In this deployment mode, the nodes are evenly distributed across the three AZs.

Figure 3-3 Specifications and storage

Table 3-3 Specifications and storage

Parameter	Description
Instance Specifications	The CPU and memory of a DB instance. Different performance specifications support different numbers of connections and maximum IOPSs. Select CPU and memory specifications based on your service requirements.
Storage Space	Storage space depends on the instance specifications. The minimum storage space is 100 GB, and the storage space you set must be an integer.

Figure 3-4 Network and database configuration

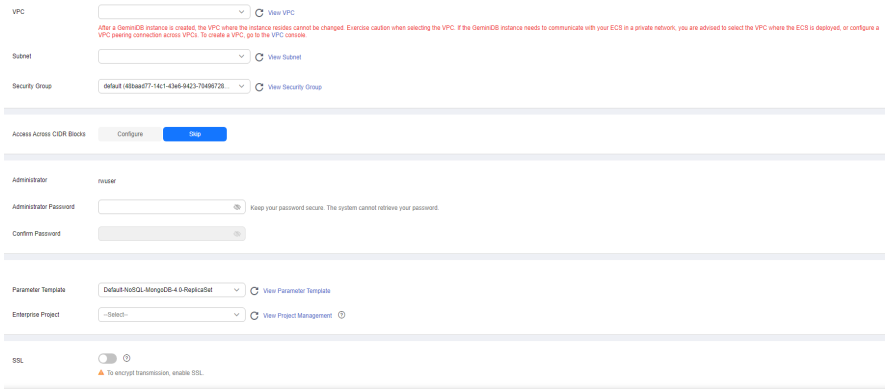


Table 3-4 Network

Parameter	Description
VPC	The virtual network where your DB instances are located. A VPC isolates networks for different services. You can select an existing VPC or create a VPC. For details about how to create a VPC, see section "Creating a VPC" in the *Virtual Private Cloud User Guide*. With VPC sharing, you can also use a VPC and subnet shared by another account. VPC owners can share the subnets in a VPC with one or multiple accounts through Resource Access Manager (RAM). This allows for more efficient use of network resources and reduces O&M costs. For more information about VPC subnet sharing, see **VPC Sharing** in the *Virtual Private Cloud User Guide*. If there are no VPCs available, the system allocates resources to you by default. **NOTE** • After the GeminiDB Mongo instance is created, the VPC where the instance resides cannot be changed. • If you want to connect to a GeminiDB Mongo instance through an ECS over an internal network, the instance and the ECS must be in the same VPC. If they are not in the same VPC, you can create a **VPC peering connection** to enable access.
Subnet	A subnet provides dedicated network resources that are logically isolated from other networks, improving network security. **NOTE** IPv6 subnets are not supported. You are advised to create and select IPv4 subnets.

Parameter	Description
Security Group	A security group controls access between GeminiDB Mongo instances and other services. When you select a security group, you must ensure that it allows the client to access DB instances. If there are no security groups available, the system allocates resources to you by default.
SSL	Secure Sockets Layer (SSL) encrypts connections between clients and servers, preventing data from being tampered with or stolen during transmission. You can enable SSL to improve data security. After an instance is created, you can connect to it using **SSL**. **NOTE** • This function is in the open beta test (OBT) phase. To use this function, contact customer service. • If SSL is not enabled when you create an instance, you can enable it after the instance is created. For details, see **Configuring an SSL Connection**.
Access Across CIDR Blocks	Access across CIDR blocks is required only when the CIDR blocks of the client and the replica set instance are different. For example, if the client CIDR block is 192.168.0.0/16 and the replica set instance's CIDR block is 172.16.0.0/24, add the CIDR block 192.168.0.0/16 so that the client can access the replica set instance. You can configure cross-CIDR block access when creating a DB instance or after the instance is created. • **Configure** When creating a DB instance, add the CIDR block of the source client (for example, the client that accesses the replica set instance). After the DB instance is created, you can connect to the DB instance from the source client. If there are multiple CIDR blocks, click ⊕ to add them in sequence. Up to 9 CIDR blocks can be configured, but they cannot overlap. **NOTE** – To ensure the ECS and the DB instance can communicate with each other, configure the connection by referring to **VPC Peering Connection Overview**. • **Skip** Configure the CIDR block of the client later. After a DB instance is created, you can configure access across CIDR blocks by referring to **Configuring Access Across CIDR Blocks**.

Table 3-5 Database configuration

Parameter	Description
Administrator	The default administrator account is **rwuser**.
Administrator Password	Set a password for the administrator. The password: • Must be 8 to 32 characters long. • Must contain uppercase letters, lowercase letters, digits, and any of the following special characters: ~!@#%^*-_=+? • For security reasons, you must select a strong password. The system will verify the password strength. Keep this password secure. If you lose it, the system cannot retrieve it.
Confirm Password	Enter the administrator password again.
Parameter Template	A parameter template contains API configuration values that can be applied to one or more instances. After an instance is created, you can modify its parameters to better meet your service requirements. For details, see **Modifying Parameters of GeminiDB Mongo Instances**. **NOTICE** When an instance is created, the system uses default values for specification parameters in your customer parameter template.
Enterprise Project	This parameter is provided for enterprise users. An enterprise project groups cloud resources, so you can manage resources and members by project. The default project is **default**. Select an enterprise project from the drop-down list. For more information about enterprise projects, see ***Enterprise Management User Guide***.

Table 3-6 Tags

Parameter	Description
Tags	This setting is optional. Adding tags helps you better identify and manage your instances. By default, each instance supports up to 10 tags. To use more tags, contact customer service to apply for a quota of 20 tags. A tag consists of a tag key and a tag value. If your organization has configured tag policies for GeminiDB Mongo, you need to add tags to instances based on the tag policies. If a tag does not comply with the policies, DB instance creation may fail. Contact your organization administrator to learn more about tag policies. • Tag key: Mandatory if the instance is going to be tagged. Each tag key must be unique for each instance. The key can include up to 36 characters, including digits, letters, underscores (_), and hyphens (-). • Tag value: Optional if the instance is going to be tagged The value can contain up to 43 characters, including digits, letters, underscores (_), periods (.), and hyphens (-). After a DB instance is created, you can view its tag details on the **Tags** tab. In addition, you can add, modify, and delete tags for existing DB instances. For details, see **Managing Tags**.

Table 3-7 Required duration

Parameter	Description
Required Duration	The length of your subscription if you select **Yearly/Monthly** billing. Subscription lengths range from one month to three years.
Auto-renew	• By default, this option is not selected. • If you select this option, the renew cycle is the same as the selected duration.

Table 3-8 Quantity

Parameter	Description
Quantity	GeminiDB Mongo instances can be created in batches. Up to 50 DB instances can be created at a time.

Step 5 On the displayed page, confirm the instance details.

- For yearly/monthly instances
  - If you need to modify the settings, click **Previous**.
  - If you do not need to modify the settings, read and agree to the service agreement and click **Pay Now** to go to the payment page and complete the payment.
- For pay-per-use instances
  - If you need to modify the settings, click **Previous**.
  - If no modification is required, read and agree to the service agreement and click **Submit**.

Step 6 On the **Instances** page, view and manage your instances.

- Creating a DB instance takes about 5 to 9 minutes. During the process, the instance status displayed in the DB instance list is **Creating**.
- After the creation is complete, the status changes to **Available**.

  You can click  in the upper right corner of the page to refresh the DB instance statuses.
- During creation, an automated backup policy is enabled by default. A full backup is automatically triggered after a DB instance is created.

----End

3.3 Connecting to a Replica Set Instance

3.3.1 Connection Methods

You can connect to a GeminiDB Mongo replica set instance through a private or public network.

Table 3-9 Connection methods

Method	Scenario	Description
Private network	GeminiDB provides a private IP address by default. Your applications are deployed on an ECS that is in the same region and VPC as your instances.	High security and performance

Method	Scenario	Description
Public network	If you cannot access a DB instance through a private IP address, bind an EIP to the DB instance first and connect the ECS to the DB instance through the EIP.	• Low security • For faster transmission and improved security, you are advised to migrate your applications to an ECS that is in the same subnet as your instance and use a private IP address to access the instance.

3.3.2 Connecting to a Replica Set Instance over a Private Network

If your applications are deployed on an ECS that is in the same region and VPC as your instances, you can connect to the DB instance over a private network.

This section describes how to use the MongoDB client to connect to a GeminiDB Mongo replica set instance over a private network.

The following uses a Linux ECS and a Window client as an example.

The MongoDB client can connect to an instance with an unencrypted connection or an encrypted connection (SSL). To secure data transmission, connect to your instances using SSL.

Precautions

- The instances must be in the same VPC and subnet as the ECS.
- The ECS must be in a security group that has access to the instances.
  - If the instance is associated with the default security group, you do not need to configure security group rules.
  - If the instance is not associated with the default security group, check whether the security group rules allow the ECS to connect to the instance.

    If yes, the ECS can connect to the instance.

    If no, add an inbound rule to the security group. For details, see **Configuring Security Group Rules**.

Prerequisites

1. You have created and logged in to an ECS. For details, see **Purchasing an ECS** and **Logging In to an ECS**.
2. You have installed the MongoDB client on the ECS.

   For details about how to install the MongoDB client, see **How Can I Install a MongoDB Client?**

SSL Connection

NOTICE

If you connect to an instance over the SSL connection, enable SSL first. Otherwise, an error is reported. For details about how to enable SSL, see **Configuring an SSL Connection**.

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, click the instance.

Step 4 In the navigation pane on the left, choose **Connections**.

Step 5 In the **Basic Information** area, click next to the **SSL** field to download the root certificate.

Step 6 Upload the root certificate to the ECS to be connected to the instance.

The following describes how to upload the certificate to a Linux and a Window ECS

- In Linux, run the following command:

  scp *<IDENTITY_FILE>* *<REMOTE_USER>***@***<REMOTE_ADDRESS>***:***<REMOTE_DIR>*

  NOTE

  - **IDENTITY_FILE** is the directory where the root certificate resides. The file access permission is 600.
  - **REMOTE_USER** is the ECS OS user.
  - **REMOTE_ADDRESS** is the ECS address.
  - **REMOTE_DIR** is the directory of the ECS to which the root certificate is uploaded.
- In Windows, upload the root certificate using a remote connection tool.

Step 7 Connect to a DB instance.

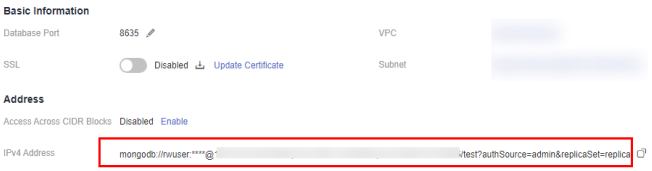
- **Method 1**: Use the connection address to connect to a replica set instance (recommended).

  The connection addresses provided on the GeminiDB Mongo console includes the IP addresses and ports of the primary node, standby node, and hidden node by default. You can use this connection address to connect to the primary, standby, and hidden nodes of the instance at the same time. In this way, write errors can be avoided after a primary/standby switchover.

  Example command:

  ./mongo "*<Connection address>***" --ssl --sslCAFile** *<FILE_PATH>* **--sslAllowInvalidHostnames**

Table 3-10 Parameter description

Parameter	Description
<Connection address>	Connection address of the instance to be connected. On the **Instance Management** page, click the instance to go to the **Basic Information** page. Obtain the private IP address on the **Connections** page. **Figure 3-5** Obtaining the connection address  – In the preceding command, ***** needs to be replaced with the password of the instance administrator. If the password contains at signs (@), exclamation marks (!), or percent signs (%), replace them with hexadecimal URL codes %40, %21, and %25 respectively. For example, if the password is ****@ %***!, the corresponding URL code is **** %40%25*** %21. – **replica** in **replicaSet=replica** is the name of a replica set. The name of GeminiDB Mongo replica set is fixed and cannot be changed.
<FILE_PATH>	Path for storing the root certificate.

Example:

./mongo "mongodb://rwuser:**@192.168.0.252:8635,192.168.0.31:8635/test?authSource=admin&replicaSet=replica" --ssl --sslCAFile/tmp/ca.crt --sslAllowInvalidHostnames**

NOTE

- A replica set instance uses the management IP address to generate SSL certificate. **--sslAllowInvalidHostnames** is required when you connect to a replica set over a private network.
- If you connect to an instance over a private IP address, add double quotation marks before and after the connection information.

If the following information is displayed, the instance is successfully connected:

```
replica:PRIMARY>
```

- **Method 2**: Connect to a single node.

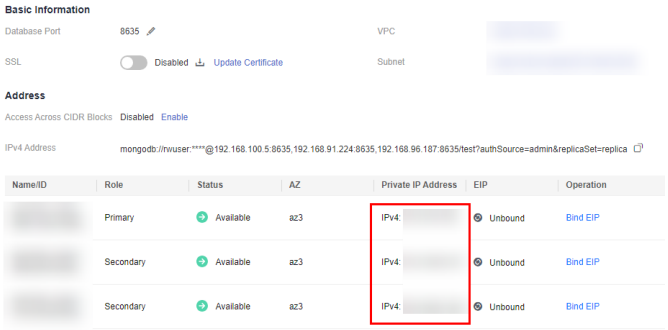
  Example command:

./mongo --host <*DB_HOST*> **--port** <*DB_PORT*> **-u** <*DB_USER*> **-p --authenticationDatabase admin --ssl --sslCAFile**<*FILE_PATH*> **--sslAllowInvalidHostnames**

Enter the password of the database account if the following information is displayed:

```
Enter password:
```

Table 3-11 Parameter description

Parameter	Description
<*DB_HOST*>	Private IP address of the primary or standby node of the instance to be connected. Primary node: You can read and write data. Secondary node: You can only read data. On the **Instance Management** page, click the instance to go to the **Basic Information** page. On the **Connections** page, obtain the private IP address of the corresponding node. **Figure 3-6** Obtaining the node IP address  Alternatively, obtain the private IP address of a node in the **Node Information** area on the **Basic Information** page. **Figure 3-7** Obtaining the private IP address 
<*DB_PORT*>	Database port
<*DB_USER*>	Username of the instance administrator. The default value is **rwuser**.
<*FILE_PATH*>	Path for storing the root certificate.

Example:

./mongo --host 192.168.0.31 --port 8635 -u rwuser -p --authenticationDatabase admin --ssl --sslCAFile /tmp/ca.crt --sslAllowInvalidHostnames

If the following information is displayed, the corresponding node is successfully connected:

- The primary node of the replica set is connected.
  ```
  replica:PRIMARY>
  ```
- The standby node of the replica set is connected.
  ```
  replica:SECONDARY>
  ```

----End

Non-SSL Connection

NOTICE

If you connect to an instance over an unencrypted connection, disable SSL first. Otherwise, an error is reported. For details about how to disable SSL, see **Configuring an SSL Connection**.

Step 1 Connect to the ECS.

Step 2 Connect to the DB instance in the directory where the MongoDB client is located.

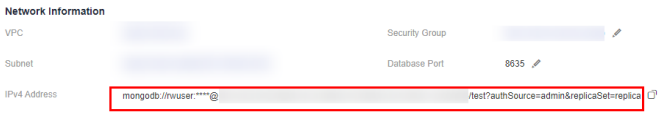
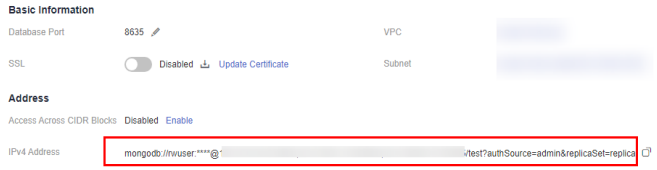
- Use the connection address to connect to a replica set instance (recommended).

  The connection addresses provided on the GeminiDB Mongo console includes the IP addresses and ports of the primary node, standby node, and hidden node by default. Connecting to a replica set with this method allows access to the primary, standby, and hidden nodes at the same time, so write errors can be avoided after a primary/standby switchover.

  Example command:

  ./mongo "*<Connection address>*"

Table 3-12 Parameter description

Parameter	Description
<Connection address>	You can obtain the connection address in either of the following ways: Click the instance name. On the displayed **Basic Information** page, the address is displayed in the **Address** field in the **Network Information** area. **Figure 3-8** Obtaining the private IP address  Alternatively, click the instance name to go to the **Basic Information** page. In the navigation pane on the left, choose **Connections** to obtain the address of the instance. **Figure 3-9** Obtaining the connection address  – In the preceding command, ***** needs to be replaced with the password of the instance administrator. If the password contains at signs (@),exclamation marks (!), or percent signs (%), replace them with hexadecimal URL codes %40, %21, and %25 respectively. For example, if the password is ****@ %***!, the corresponding URL code is **** %40%25*** %21. – **replica** in **replicaSet=replica** is the name of a replica set. The name of GeminiDB Mongo replica set is fixed and cannot be changed.

Example:

./mongo "mongodb://rwuser:**@192.168.0.196:8635,192.168.0.67:8635,192.168.0.32:8635/test?authSource=admin&replicaSet=replica"**

If the following information is displayed, the connection is successful.

```
replica:PRIMARY>
```

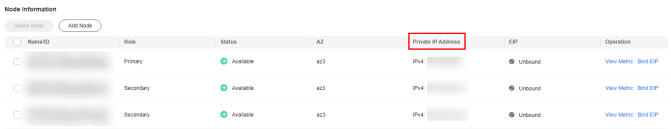
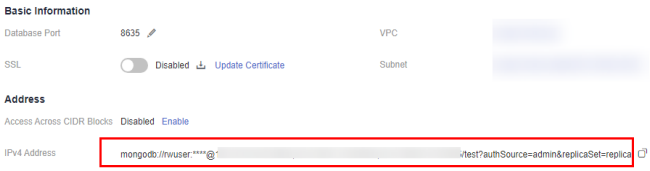
- Connecting to a single node.

  You can also connect to the private IP address of a specified node.

  Example command:

./mongo --host <*DB_HOST*> **--port** <*DB_PORT*> **-u** <*DB_USER*> **-p --authenticationDatabase admin**

Table 3-13 Parameter description

Parameter	Description
<*DB_HOST*>	Private IP address of the primary or standby node of the instance to be connected. – Primary node: You can read and write data. – Secondary node: You can only read data. On the **Instance Management** page, click the instance to go to the **Basic Information** page. Obtain the private IP address of the corresponding node from the node information list. **Figure 3-10** Obtaining the private IP address  Alternatively, click the target instance to go to the **Basic Information** page. In the navigation pane on the left, choose **Connections**. On the **Connections** page, obtain the private IP address of the node. **Figure 3-11** Obtaining the connection address 
<*DB_PORT*>	Database port
<*DB_USER*>	Username of the instance administrator. The default value is **rwuser**.

Enter the password of the database account if the following information is displayed:

```
Enter password:
```

Example:

./mongo --host 192.168.1.6 --port 8635 -u rwuser -p --authenticationDatabase admin

If the following information is displayed, the corresponding node is successfully connected:

The command output for connecting to the primary node of the replica set is as follows:

```
replica:PRIMARY>
```

The command output for connecting to the standby node of the replica set is as follows:

```
replica:SECONDARY>
```

----End

3.3.3 Connecting to a Replica Set Instance over a Public Network

3.3.3.1 Connecting to a Replica Set Instance Using Robo 3T

If you connect to an instance from your local device, you can bind an EIP to the instance and use Robo 3T to connect to the instance over a public network.

This section describes how to use Robo 3T to connect to a replica set instance over a public network.

A local Windows device is used as an example.

Robo 3T can connect to an instance with an unencrypted connection or an encrypted connection (SSL). To improve data transmission security, you are advised to connect to instances using the SSL connection.

Prerequisites

1. An EIP has been bound to a replica set instance, and security group rules have been configured to ensure that the instance is accessible from Robo 3T over the EIP. For details, see **Binding and Unbinding an EIP** and **Configuring Security Group Rules**.
2. Install Robo 3T.

   For details, see **How Do I Install Robo 3T?**

SSL Connection

NOTICE

If you connect to an instance over the SSL connection, enable SSL first. Otherwise, an error is reported. For details about how to enable SSL, see **Configuring an SSL Connection**.

Step 1 Obtain the EIP and port of the replica set instance.

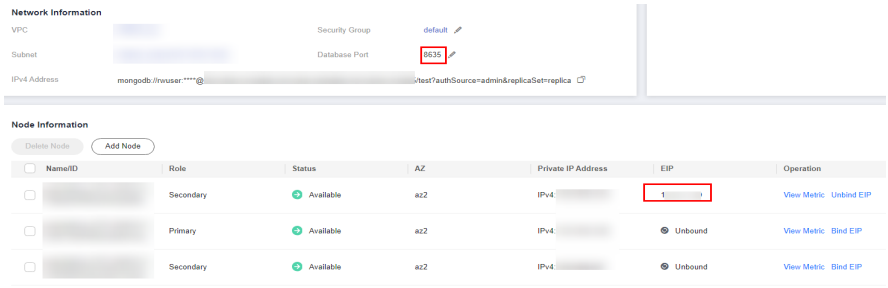
NOTE

Primary node: You can read and write data.

Secondary node: You can only read data.

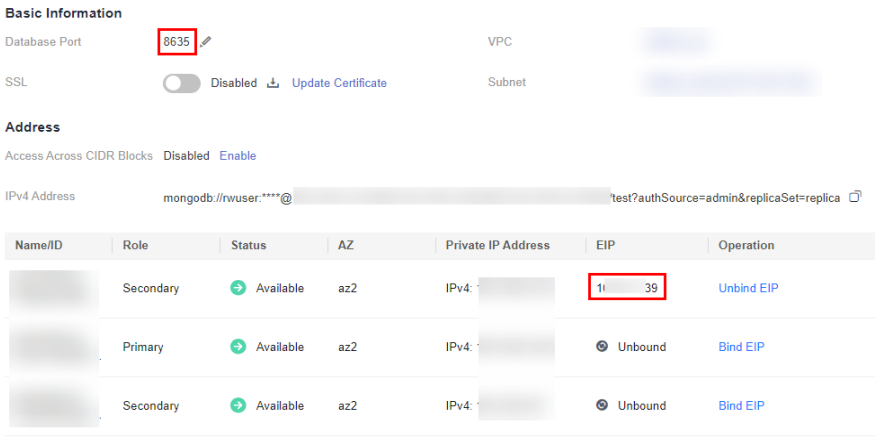
- You can click the instance name to go to the **Basic Information** page. In the **Network Information** area, the port is displayed in the **Database Port** field and the node EIPs are displayed in **Node Information** area.

  Figure 3-12 Obtaining connection information



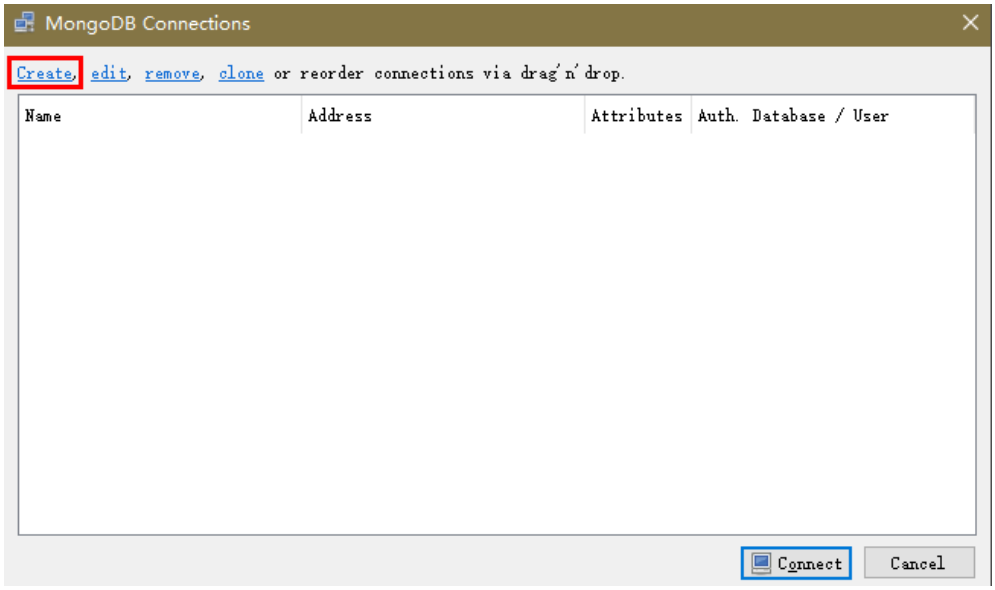
- Alternatively, click the target instance. On the displayed **Basic Information** page, click **Connections** in the navigation pane on the left. Obtain the port and EIP on the **Connections** page.

  Figure 3-13 Viewing EIPs



Step 2 Run the installed Robo 3T. On the displayed dialog box, click **Create**.

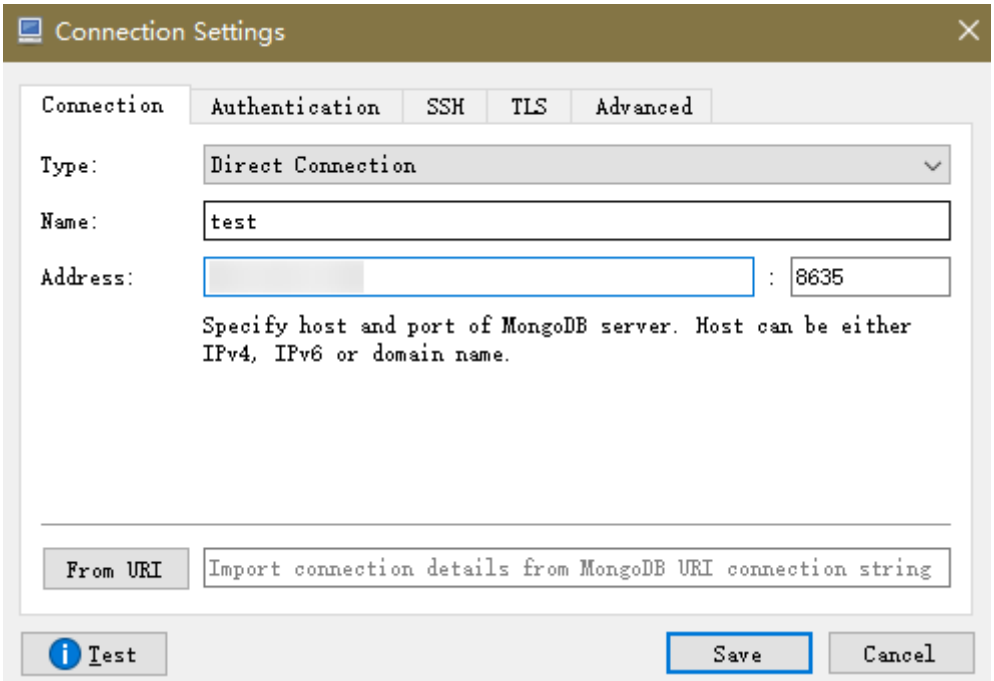
Figure 3-14 Connections



Step 3 In the **Connection Settings** dialog box, set the parameters of the new connection.

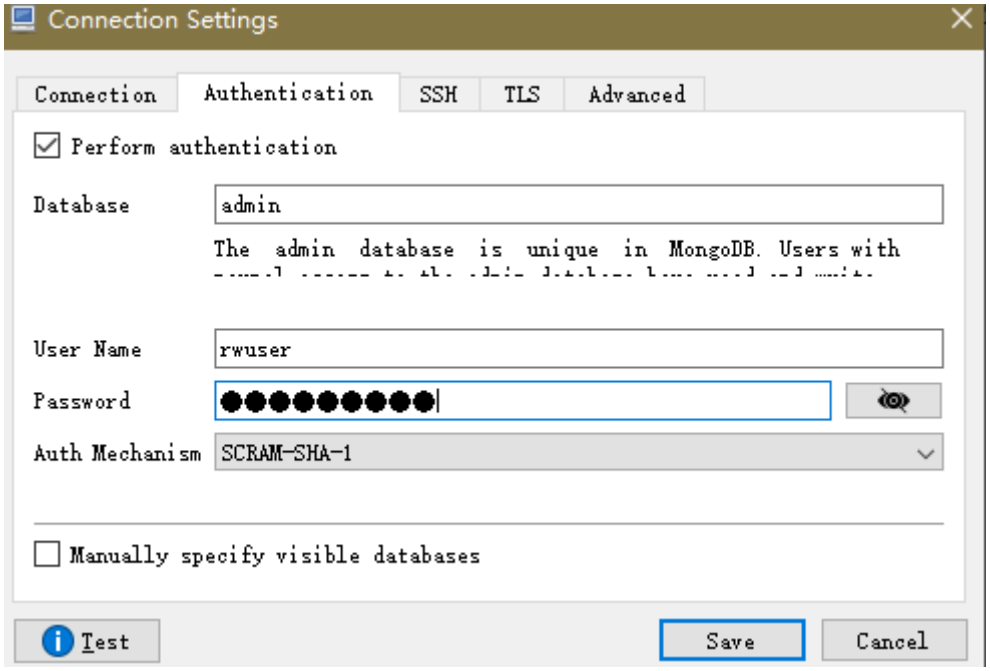
1. On the **Connection** tab, enter the name of the new connection in the **Name** text box and enter the EIP and database port that are bound to the replica set instance in the **Address** text box.

   Figure 3-15 Connection



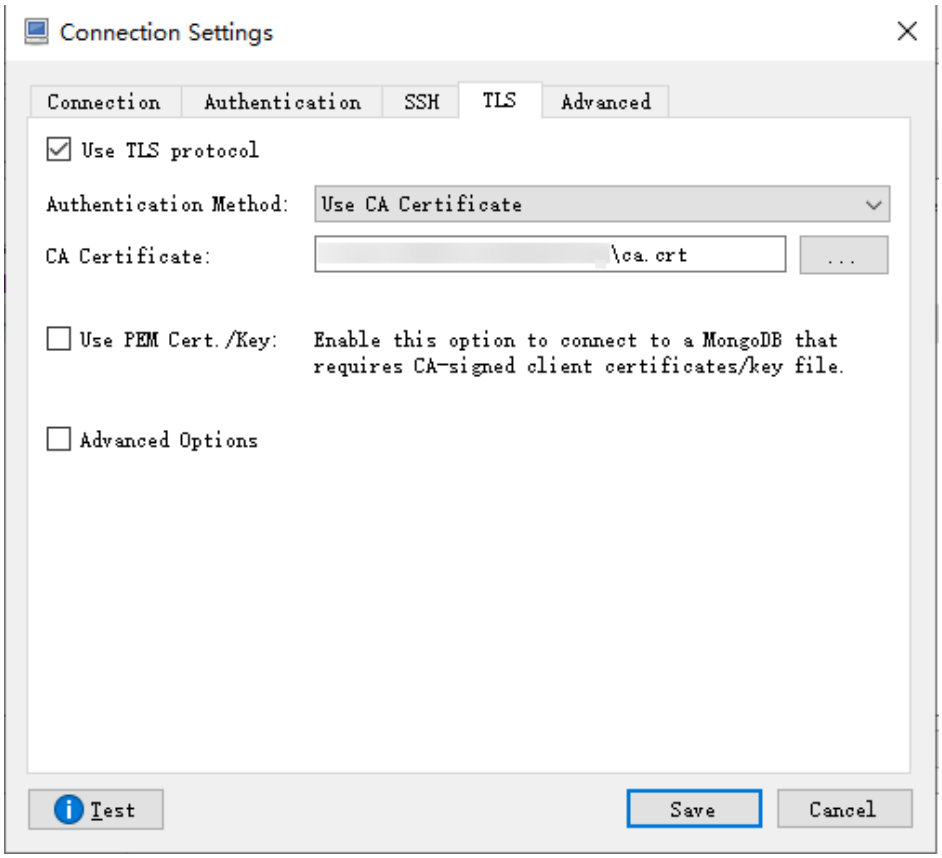
2. On the **Authentication** tab, set **Database** to **admin**, **User Name** to **rwuser**, and **Password** to the administrator password you set during the creation of the replica set instance.

Figure 3-16 Authentication



3. Click the **TLS** tab, select **Use TLS protocol**, set **Authentication Method** to **Use CA Certificate**, and upload the obtained SSL certificate.

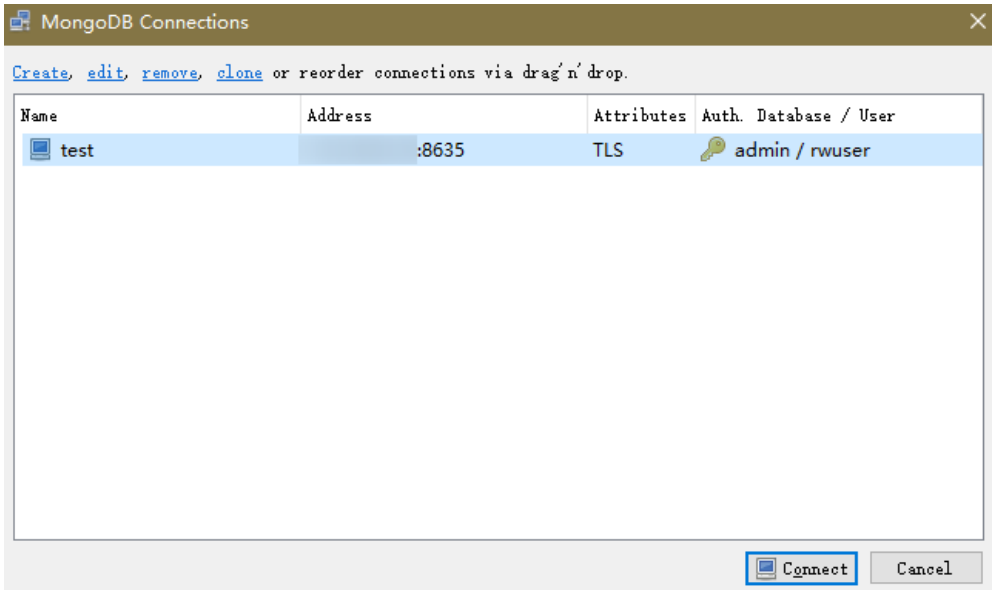
Figure 3-17 SSL



4. Click **Save**.

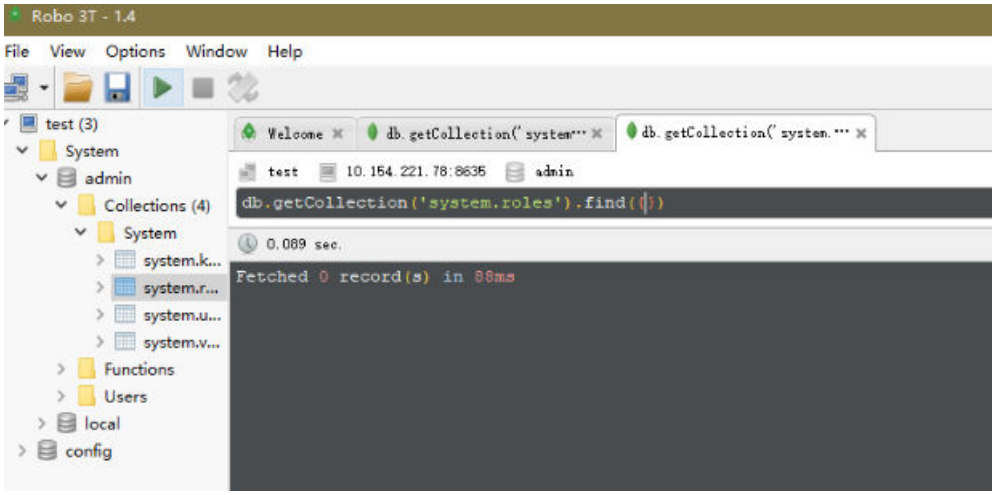
Step 4 On the **MongoDB Connections** page, click **Connect** to connect to the replica set instance.

Figure 3-18 Replica set connection information



Step 5 If the replica set instance is successfully connected, the following page is displayed.

Figure 3-19 Connection succeeded



----End

Non-SSL Connection

NOTICE

If you connect to an instance over an unencrypted connection, disable SSL first. Otherwise, an error is reported. For details about how to disable SSL, see **Configuring an SSL Connection**.

Step 1 Obtain the EIP and port of the replica set instance.

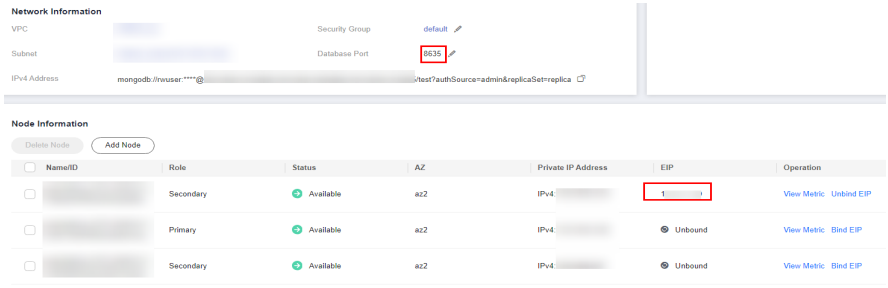
NOTE

Primary node: You can read and write data.

Secondary node: You can only read data.

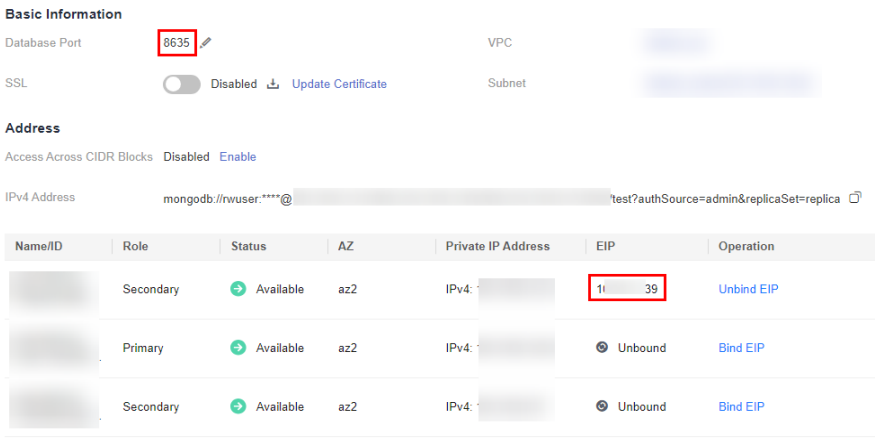
- You can click the instance name to go to the **Basic Information** page. In the **Network Information** area, the port is displayed in the **Database Port** field and the node EIPs are displayed in **Node Information** area.

Figure 3-20 Obtaining connection information



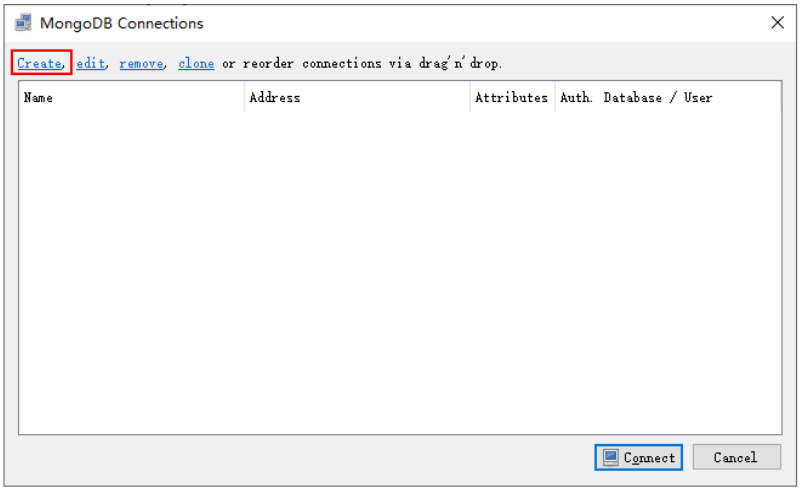
- Alternatively, click the target instance. On the displayed **Basic Information** page, click **Connections** in the navigation pane on the left. Obtain the port and EIP on the **Connections** page.

Figure 3-21 Viewing EIPs



Step 2 Run the installed Robo 3T. On the displayed dialog box, click **Create**.

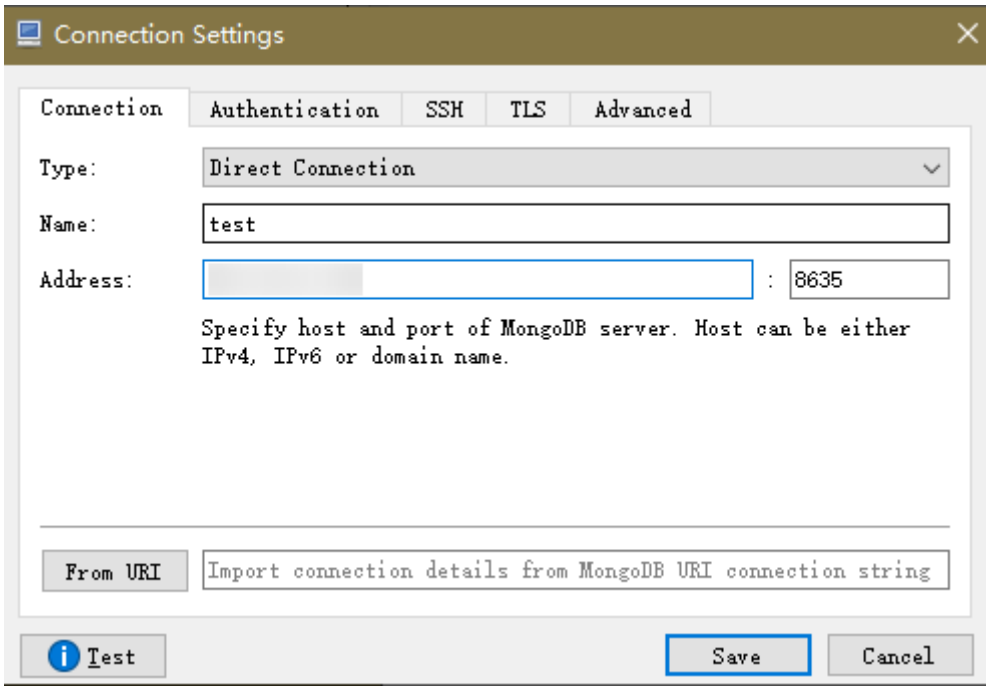
Figure 3-22 Connections



Step 3 In the **Connection Settings** dialog box, set the parameters of the new connection.

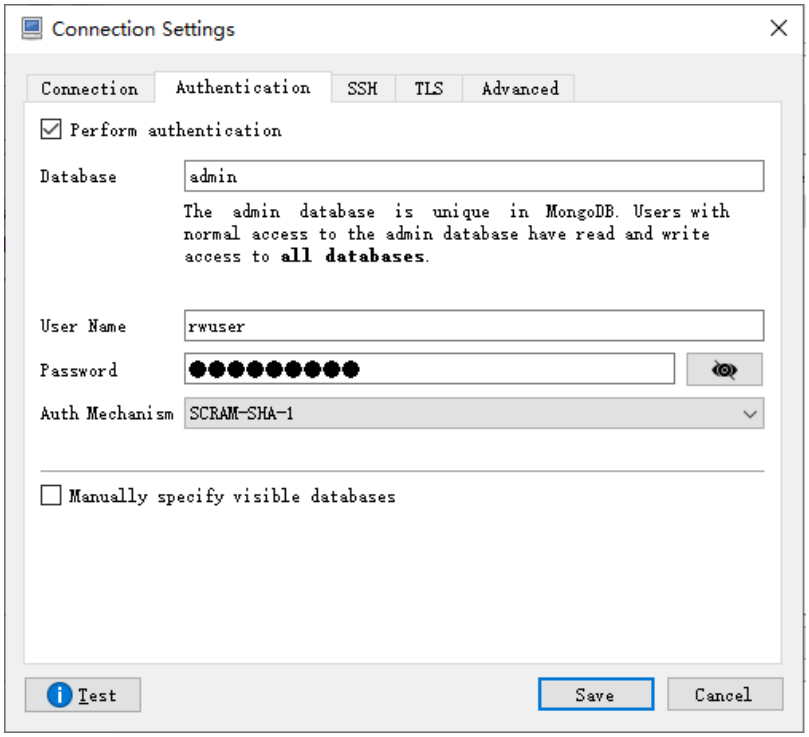
1. On the **Connection** tab, specify the connection name in **Name** and enter the EIP and port of the replica set instance obtained in **Step 1** in **Address**.

Figure 3-23 Connection



2. On the **Authentication** tab, set **Database** to **admin**, **User Name** to **rwuser**, and **Password** to the administrator password you set during the creation of the replica set instance.

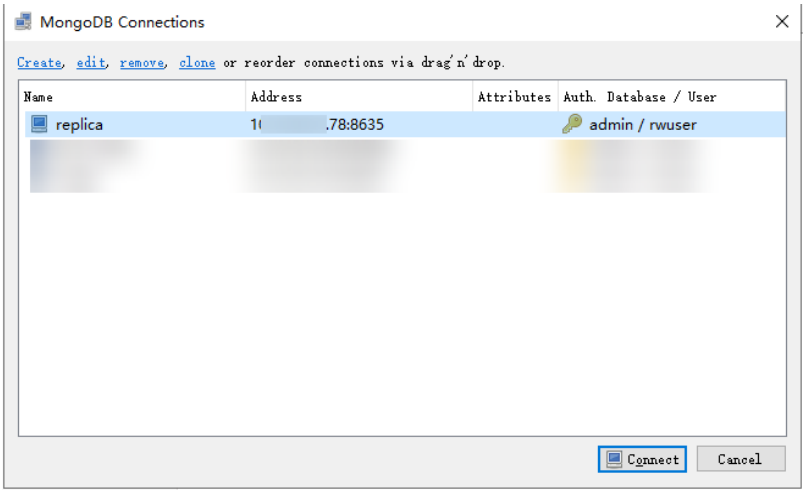
Figure 3-24 Authentication



3. Click **Save**.

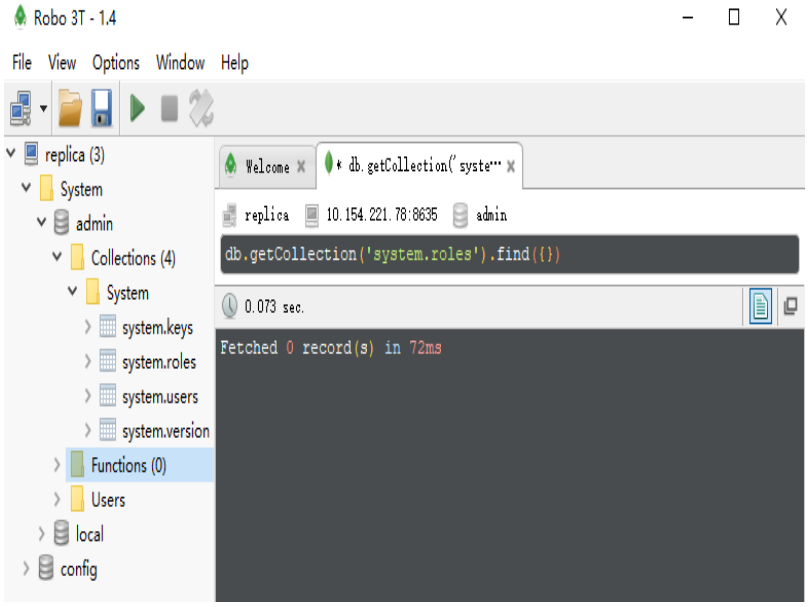
Step 4 On the **MongoDB Connections** page, click **Connect** to connect to the replica set instance.

Figure 3-25 Connections



Step 5 If the replica set instance is successfully connected, the page shown in **Figure 3-26** is displayed.

Figure 3-26 Connection succeeded



----End

3.3.3.2 Connecting to a Replica Set Instance Using the MongoDB Client

If a DB instance cannot be accessed over a private network, you can bind an EIP to the instance.

This section describes how to use the MongoDB client to connect to a replica set instance over a public network.

A Linux ECS is used as an example.

The MongoDB client can connect to an instance with an unencrypted connection or an encrypted connection (SSL). To improve data transmission security, you are advised to connect to instances using the SSL connection.

Prerequisites

1. You have bound an EIP to the instance and configured security group rules. In this way, you can access the instance from the ECS. For details, see **Binding and Unbinding an EIP** and **Configuring Security Group Rules**.
2. Install the MongoDB client.
   a. You have created and logged in to an ECS. For details, see **Purchasing an ECS** and **Logging In to an ECS**.
   b. You have installed the MongoDB client on the ECS.

      For details about how to install the MongoDB client, see **How Can I Install a MongoDB Client?**

SSL Connection

NOTICE

If you connect to an instance over the SSL connection, enable SSL first. Otherwise, an error is reported. For details about how to enable SSL, see **Configuring an SSL Connection**.

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, click the instance.

Step 4 In the navigation pane on the left, choose **Connections**.

Step 5 In the **Basic Information** area, click next to the **SSL** field to download the root certificate.

Step 6 Upload the root certificate to the ECS to be connected to the instance.

The following describes how to upload the certificate to a Linux and a Window ECS

- In Linux, run the following command:

  scp *<IDENTITY_FILE><REMOTE_USER>***@***<REMOTE_ADDRESS>***:***<REMOTE_DIR>*

  NOTE

  - **IDENTITY_FILE** is the directory where the root certificate resides. The file access permission is 600.
  - **REMOTE_USER** indicates the OS user.
  - **REMOTE_ADDRESS** indicates the host address.
  - **REMOTE_DIR** indicates the directory to which the root certificate is uploaded.
- In Windows, upload the root certificate using a remote connection tool.

Step 7 Connect to the DB instance in the directory where the MongoDB client is located.

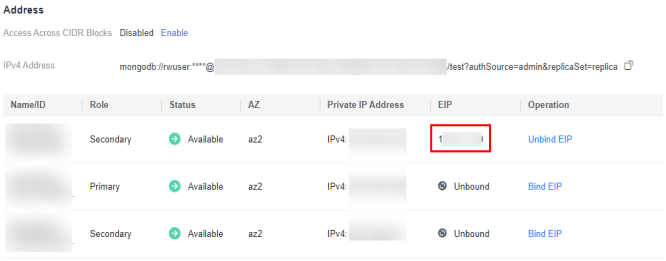
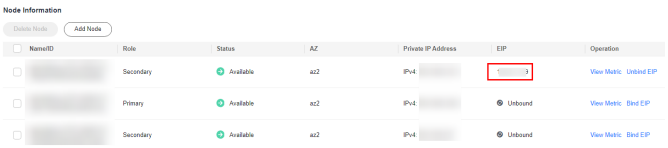
Example command:

./mongo --host *<DB_HOST>* **--port** *<DB_PORT>* **-u** *<DB_USER>* **-p --authenticationDatabaseadmin --ssl --sslCAFile***<FILE_PATH>* **--sslAllowInvalidHostnames**

Enter the password of the database account if the following information is displayed:

```
Enter password:
```


Table 3-14 Parameter description

Parameter	Description
<*DB_HOST*>	EIP bound to the instance node to be connected. Primary node: You can read and write data. Secondary node: You can only read data. On the **Instance Management** page, click the instance to go to the **Basic Information** page. On the **Connections** page, obtain the EIP of the corresponding node. **Figure 3-27** Obtaining the EIP of a Node  Alternatively, click the instance name and obtain the EIP of the corresponding node in the **Node Information** area. **Figure 3-28** Viewing EIPs 
<*DB_PORT*>	Database port
<*DB_USER*>	Username of the instance administrator. The default value is **rwuser**.
<*FILE_PATH*>	Path for storing the root certificate.

Example:

./mongo --host 192.168.1.6 --port 8635 -u rwuser -p --authenticationDatabase admin --ssl --sslCAFile /tmp/ca.crt --sslAllowInvalidHostnames

Step 8 Check the results. If the following information is displayed, the connection is successful.

- The primary node of the replica set is connected.
  ```
  replica:PRIMARY>
  ```
- The standby node of the replica set is connected.


```
replica:SECONDARY>
```

----End

Non-SSL Connection

NOTICE

If you connect to an instance over an unencrypted connection, disable SSL first. Otherwise, an error is reported. For details about how to disable SSL, see **Configuring an SSL Connection**.

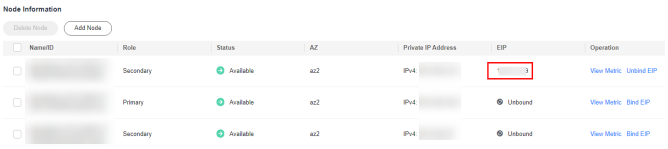
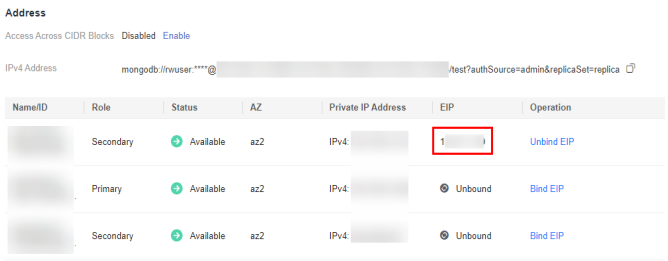
Step 1 Connect the ECS.

Step 2 Connect to the instance.

Example command:

./mongo --host <*DB_HOST*> **--port** <*DB_PORT*> **-u** <*DB_USER*> **-p --authenticationDatabase admin**

Table 3-15 Parameter description

Parameter	Description
<DB_HOST>	EIP of the primary or standby node of the instance to be connected. • Primary node: You can read and write data. • Secondary node: You can only read data. On the **Instance Management** page, click the instance to go to the **Basic Information** page. Obtain the EIP of the corresponding node from the node information list. **Figure 3-29** Obtaining an EIP  Alternatively, click the target instance to go to the **Basic Information** page. In the navigation pane on the left, choose **Connections**. On the **Connections** page, obtain the EIP of the node. **Figure 3-30** Obtaining the EIP of a node 
<DB_PORT>	Database port
<DB_USER>	Username of the instance administrator. The default value is **rwuser**.

Enter the password of the database account if the following information is displayed:

```
Enter password:
```

Example:

./mongo --host 192.168.1.6 --port 8635 -u rwuser -p --authenticationDatabase admin

Step 3 Check the results. If the following information is displayed, the connection is successful.

- The primary node of the replica set is connected.
  ```
  replica:PRIMARY>
  ```
- The standby node of the replica set is connected.
  ```
  replica:SECONDARY>
  ```

----End

3.4 Getting Started with Common Practices

After purchasing and connecting to an instance, you can view common practices to better use GeminiDB Mongo.

Table 3-16 Common practices

Practice		Description
Usage rules	**Naming Rules**	This section describes the naming conventions of GeminiDB Mongo API.
	Indexing Rules	This section describes the index specifications of GeminiDB Mongo API.
Instance modifications	**Modifying the Name of an Instance**	This section describes how to change an instance name to identify different instances.
	Resetting the Administrator Password	For security reasons, regularly change your administrator password.
	Changing Specifications of an Instance	Describes how to change the CPU or memory of your instance to suit your service requirements.
Data backup	**Managing Automated Backups**	Describes how GeminiDB Mongo API automatically creates backups for a DB instance during a backup window and saves the backups based on the configured retention period.
	Managing Manual Backups	Describes how to create manual backups for a DB instance. These backups can be used to restore data for improved reliability.
Data restoration	**Restoring Data to a New or Existing Instance**	This practice describes how to restore an existing automated or manual backup to a new or existing instance. The restored data is the same as the backup data.
Log management	**Slow Query Logs**	GeminiDB Mongo API allows you to view slow query logs of databases. The unit of the execution time is ms. You can identify the SQL statements that take a long time to execute and tune them based on slow query logs.

Practice		Description
	Error Logs	GeminiDB Mongo API allows you to view error ogs of databases, including warning logs and error logs generated during database running, helping you analyze system issues.

4 Working with GeminiDB Mongo API

4.1 Permissions Management

4.1.1 Creating a User Group and Assigning Permissions

This section describes how to use **IAM** to control fine-grained permissions for your GeminiDB resources. With IAM, you can:

- Create IAM users for employees based on your enterprise's organizational structure. Each IAM user will have their own security credentials for accessing GeminiDB resources.
- Grant only the permissions required for users to perform a specific task.
- Entrust a Huawei Cloud account or cloud service to perform efficient O&M on your GeminiDB resources.

If your Huawei Cloud account does not require individual IAM users, skip this section.

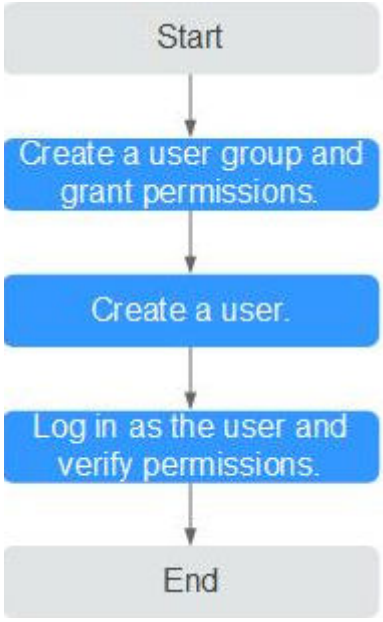
The following describes the procedure for granting permissions (see **Figure 4-1**).

Prerequisites

Learn about the permissions supported by GeminiDB and choose policies or roles based on your requirements. For details about the permissions, see **Permissions Management**. For system policies of other services, see **Permissions Policies**.

Process Flow

Figure 4-1 Process of granting GeminiDB permissions



1. **Create a user group and assign permissions** to it.

   Create a user group on the IAM console and attach the **GeminiDB FullAccess** policy to the group.

2. **Create an IAM user**.

   Create a user on the IAM console and add the user to the group created in **1**.

3. **Log in** and verify permissions.

   Log in to the management console using the created user, and verify that the user only has read permissions.

   Choose **Service List** > **GeminiDB** and click **Buy DB Instance**. If you can buy an instance, the required permission policy has taken effect.

4.1.2 Creating a Custom Policy

Custom policies can be created to supplement the system-defined policies of GeminiDB. For the actions supported for custom policies, see **Permissions Policies and Supported Actions**.

You can create custom policies in either of the following ways:

- Visual editor: Select cloud services, actions, resources, and request conditions. This does not require knowledge of policy syntax.
- JSON: Edit JSON policies from scratch or based on an existing policy.

For details, **Creating a Custom Policy**. The following contains examples of common GeminiDB custom policies.

Example Custom Policy

- Example 1: Allowing users to create GeminiDB instances

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "nosql:instance:create"
         ]
      }
   ]
}
```

- Example 2: Deny users the permission to delete GeminiDB instances.

  A policy with only "Deny" permissions must be used in conjunction with other policies to take effect. If the policies assigned to a user contain both Allow and Deny actions, the Deny actions take precedence over the Allow actions.

  The following method can be used if you need to assign permissions of the **GeminiDB FullAccess** policy to a user but you want to prevent the user from deleting GeminiDB instances. Create a custom policy for denying GeminiDB instance deletion, and attach both policies to the group to which the user belongs. Then, the user can perform all operations on GeminiDB instances except deleting GeminiDB instances. The following is an example of the deny policy:

```
{
   "Version": "1.1",
   "Statement": [
      {
        "Effect": "Deny"
        "Action": [
            "nosql:instance:delete"
         ],
       }
   ]
}
```

- Example 3: Defining permissions for multiple services in a policy

  A custom policy can contain the actions of multiple services that are of the global or project-level type. The following is an example policy containing actions of multiple services:

```
{
      "Version": "1.1",
      "Statement": [
            {
                  "Action": [
                        "nosql:instance:create",
                        "nosql:instance:rename",
                        "nosql:instance:delete",
                        "vpc:publicIps:list",
                        "vpc:publicIps:update"
                  ],
                  "Effect": "Allow"
            }
      ]
}
```


4.2 Migrating Data

4.2.1 Migration Scheme Overview

GeminiDB Mongo provides multiple migration schemes to migrate MongoDB databases in different service scenarios.

Table 4-1 Migration schemes

Scenario	Migration Type	References
Migrating MongoDB databases using export and import tools	Full	• **Migrating Data Using mongoexport and mongoimport** • **Migrating Data Using mongodump and mongorestore**
Migrating MongoDB databases using DRS	Full +incremental	**Migrating Data Using DRS**

4.2.2 Migrating Data Using DRS

Data Replication Service (DRS) helps migrate your databases to GeminiDB Mongo instances. During the database migration, the source remains operational even if a transfer is interrupted, thereby minimizing application downtime.

For details, see **Real-Time Migration**.

4.2.3 Migrating Data Using mongoexport and mongoimport

mongoexport and mongoimport are backup and restoration tools provided by the MongoDB client. You can install a MongoDB client on the local device or ECS and use the mongoexport and mongoimport tools to migrate your on-premises MongoDB databases or other cloud MongoDB databases to GeminiDB Mongo instances.

Before migrating data from a MongoDB database to GeminiDB Mongo, transfer data to a .json file using the mongoexport tool. Then, log in to the ECS or a device that can access GeminiDB Mongo and import data to GeminiDB Mongo using mongoimport.

Precautions

- mongoexport and mongoimport support only full migration. To ensure data consistency, stop services on the source database and stop writing data to the source database before the migration.
- You are advised to perform the migration during off-peak hours to avoid the impact of migration on your services.

- The admin and local system databases cannot be migrated.
- Make sure that no service set has been created in the system databases admin and local in the source database. If there is already a service set, migrate them out of the system databases admin and local before migration.
- Before importing data, ensure that the necessary indexes are there on the source database. Delete any unnecessary indexes and create any necessary indexes before migration.
- If you choose to migrate a sharded cluster, you must create a set of shards in the destination database and configure sharding. In addition, indexes must be created before migration.

Prerequisites

1. An ECS or a device that can access GeminiDB Mongo is ready for use.
   - To connect to a GeminiDB Mongo instance over a private network from an ECS, create and log in to the ECS. For details, see **Purchasing an ECS** and **Logging In to an ECS**
   - To connect to an instance through an EIP, you must:
     i. For details on how to bind an EIP, see **Binding and Unbinding an EIP**.
     ii. Ensure that your local device can access the EIP.
2. A migration tool has been installed on the prepared ECS.

   Install the migration tool. For details, see **How Can I Install a MongoDB Client?**

   NOTE

   The MongoDB client provides the mongoexport and mongoimport tools.

Exporting Data

Step 1 Log in to the ECS or the device that can access GeminiDB Mongo API.

Step 2 Use the mongoexport tool to transfer data from the source database to a .json file.

An SSL connection is used as an example. If you select an unencrypted connection, delete **--ssl --sslAllowInvalidCertificates** from the following command.

./**mongoexport --host** *<DB_ADDRESS>* **--port** *<DB_PORT>* **--ssl --sslAllowInvalidCertificates --type json --authenticationDatabase** *<AUTH_DB>* **-u** *<DB_USER>* **--db** *<DB_NAME>* **--collection** *<DB_COLLECTION>* **--out** *<DB_PATH>*

Table 4-2 Parameter description

Parameter	Description
DB_ADDRESS	Database address
DB_PORT	Database port

Parameter	Description
AUTH_DB	Database that stores *DB_USER* information. Generally, the value is **admin**.
DB_USER	Database user
DB_NAME	The name of the database to be migrated.
DB_COLLECTION	The collection of the database to be migrated.
DB_PATH	Path of storing the JSON file

Enter the database administrator password when prompted:

```
Enter password:
```

The following is an example. After the command is executed, the **exportfile.json** file will be generated:

./mongoexport --host 192.168.1.21 --port 8635 --ssl --sslAllowInvalidCertificates --type json --authenticationDatabase admin -u rwuser --db test02 --collection Test --out /tmp/mongodb/export/exportfile.json

Step 3 Check the results.

If information similar to the following is displayed, the data is successfully exported. **x** is the number of exported data records.

```
exported x records
```

Step 4 Compress the exported .json file.

gzip exportfile.json

Compressing the file helps reduce the time needed to transmit all the data. The compressed file is **exportfile.json.gz**.

----End

Importing Data

Step 1 Log in to the ECS or the device that can access GeminiDB Mongo.

Step 2 Upload the data to be imported to the ECS or the device that can access GeminiDB Mongo.

Select an uploading method based on the OS you are using.

- In Linux, for example, you can use secure copy protocol (SCP):

  scp *<IDENTITY_FILE>* *<REMOTE_USER>*@*<REMOTE_ADDRESS>*:*<REMOTE_DIR>*

  - **IDENTITY_FILE** indicates the directory where the **exportfile.json.gz** file is located. The file access permission is 600.
  - **REMOTE_USER** is the ECS OS user.

  - **REMOTE_ADDRESS** is the ECS address.
  - **REMOTE_DIR** is the directory of the ECS to which the **exportfile.json.gz** file is uploaded.
- In Windows, upload **exportfile.json.gz** to the ECS using file transfer tools.

Step 3 Decompress the package.

gzip -d *exportfile.json.gz*

Step 4 Import the JSON file to the GeminiDB Mongo database.

An SSL connection is used as an example. If you select an unencrypted connection, delete **--ssl --sslAllowInvalidCertificates** from the following command.

./**mongoimport --host** <*DB_ADDRESS*> **--port** <*DB_PORT*> **--ssl --sslAllowInvalidCertificates --type json --authenticationDatabase** <*AUTH_DB*> **-u** <*DB_USER*> **--db** <*DB_NAME*> **--collection** <*DB_COLLECTION*> **--file** <*DB_PATH*>

Table 4-3 Parameter description

Parameter	Description
DB_ADDRESS	IP address of the GeminiDB Mongo instance.
DB_PORT	Database port
AUTH_DB	The database that authenticates *DB_USER*. Generally, the value is **admin**.
DB_USER	Database administrator account
DB_NAME	Database to be imported
DB_COLLECTION	Collection in the database to be imported
DB_PATH	Path of storing the JSON file

Enter the database administrator password when prompted:

```
Enter password:
```

Example:

./mongoimport --host 192.168.1.21 --port 8635 --ssl --sslAllowInvalidCertificates --type json --authenticationDatabase admin -u rwuser --db test02 --collection Test --file /tmp/mongodb/export/exportfile.json

Step 5 Check the results.

If information similar to the following is displayed, the data is successfully migrated. **x** is the number of imported data records.

```
imported x records
```

----End

4.2.4 Migrating Data Using mongodump and mongorestore

mongodump and mongorestore are backup and restoration tools provided by the MongoDB client. You can install a MongoDB client on the local device or ECS and use the mongodump and mongorestore tools to migrate your MongoDB databases or other cloud MongoDB databases to GeminiDB Mongo instances.

Precautions

- mongodump and mongorestore support only full migration. To ensure data consistency, stop services on the source database and stop writing data to the source database before the migration.
- You are advised to perform the migration during off-peak hours to avoid impacting services.
- The admin and local system databases cannot be migrated.
- Make sure that no service set has been created in the system databases admin and local in the source database. If there is already a service set, migrate them out of the system databases admin and local before migration.
- Before importing data, ensure that the necessary indexes are there on the source database. Delete any unnecessary indexes and create any necessary indexes before migration.
- If you choose to migrate a sharded cluster, you must create a set of shards in the destination database and configure sharding. In addition, indexes must be created before migration.
- If the backup using the mongodump tool fails (for example, an error is reported when the backup progress reaches 97%), you are advised to increase the VM storage space and reserve some redundant space before performing the backup again.
- User **rwuser** can only operate service database tables. You are advised to specify databases and tables to import and export only service data. Otherwise, the insufficient permission problem may occur during full import and export.

Prerequisites

1. Prepare an ECS or a device that can access GeminiDB Mongo.
   - To connect to a GeminiDB Mongo instance over a private network from an ECS, create and log in to the ECS. For details, see **Purchasing an ECS** and **Logging In to an ECS**
   - To connect to an instance through an EIP, you must:
     i. For details on how to bind an EIP, see **Binding and Unbinding an EIP**.
     ii. Ensure that your local device can access the EIP.
2. A migration tool has been installed on the prepared ECS or the device that can access GeminiDB Mongo.

   Install the migration tool. For details, see **How Can I Install a MongoDB Client?**

NOTE

- The mongodump and mongorestore tools are part of the MongoDB client installation package.
- The MongoDB client version must match the instance version. Otherwise, compatibility issues may occur.

Exporting Data

Step 1 Log in to the ECS or the device that can access GeminiDB Mongo API.

Step 2 Back up the source database data using mongodump.

- Example command for SSL connection:

  ./mongodump --host *<DB_HOST>* **--port** *<DB_PORT>* **--authenticationDatabase** *<AUTH_DB>* **-u <DB_USER> --ssl --sslCAFile** <*FILE_PATH*> **--sslAllowInvalidCertificates --db** <*DB_NAME*> **--collection** <*DB_COLLECTION*> **--gzip --archive=**<Backup_directory>

- Example command for unencrypted connection:

  ./mongodump --host *<DB_HOST>* **--port** *<DB_PORT>* **--authenticationDatabase** *<AUTH_DB>* **-u <DB_USER> --db** <*DB_NAME*> **--collection** <*DB_COLLECTION*> **--gzip --archive=**<Backup_directory>

Table 4-4 Parameter description

Parameter	Description
DB_HOST	Database address
DB_PORT	Database port
DB_USER	Database username
AUTH_DB	Database that stores *<DB_USER>* information. Generally, the value is **admin**.
FILE_PATH	Path for storing the root certificate
DB_NAME	The name of the database to be migrated.
DB_COLLECTION	Collection in the database to be migrated
Backup_directory	Path for storing the exported data. You can customize the path.

Enter the database administrator password when prompted:

```
Enter password:
```

Example:

./mongodump --host 192.168.xx.xx --port 8635 --authenticationDatabase admin -u rwuser --db test --collection usertable --gzip --archive=/tmp/usertable.tar.gz


```
[root@ecs-5602 bin]# ./mongodump --host 192.168.  '.   --port 8635  --authenticationDatabase admin  -u rwuser --db test --colle
ction usertable --gzip --archive=/tmp/usertable.tar.gz
Enter password:

2022-01-06T20:44:12.696+0800    writing test.usertable to archive '/tmp/usertable.tar.gz'
2022-01-06T20:44:14.596+0800    [........................]  test.usertable  101/100000  (0.1%)
2022-01-06T20:44:14.860+0800    [########################]  test.usertable  100000/100000  (100.0%)
2022-01-06T20:44:14.933+0800    done dumping test.usertable  (100000 documents)
```

----End

Importing Data

Step 1 Log in to the ECS or the device that can access GeminiDB Mongo.

Step 2 Upload the data to be imported to the ECS or the device that can access GeminiDB Mongo.

Select an uploading method based on the OS you are using.

- In Linux, for example, you can use secure copy protocol (SCP):

  scp -r *<IDENTITY_DIR>* *<REMOTE_USER>*@*<REMOTE_ADDRESS>*:*<REMOTE_DIR>*

  Table 4-5 Parameter description

Parameter	Description
<IDENTITY_DIR>	Directory where the backup folder is located.
<REMOTE_USER>	User of the ECS OS
<REMOTE_ADDRESS>	ECS address
<REMOTE_DIR>	Directory of the file to be imported

- In Windows, use a transfer tool to upload the backup file to the ECS.

Step 3 Import the backup data to the GeminiDB Mongo instance.

- Example command for SSL connection:

  ./mongorestore --host *<DB_HOST>* **--port** *<DB_PORT>* **--authenticationDatabase** *<AUTH_DB>* **-u** *<DB_USER>* **--ssl --sslCAFile** *<FILE_PATH>* **--sslAllowInvalidCertificates --db** *<DB_NAME>* **--collection** *<DB_COLLECTION>* **--gzip --archive=***<Backup_directory>*

- Example command for unencrypted connection:

  ./mongorestore --host *<DB_HOST>* **--port** *<DB_PORT>* **--authenticationDatabase** *<AUTH_DB>* **-u** *<DB_USER>* **--db** *<DB_NAME>* **--collection** *<DB_COLLECTION>* **--gzip --archive=***<Backup_directory>*

Table 4-6 Parameter description

Parameter	Description
DB_HOST	Database address
DB_PORT	Database port

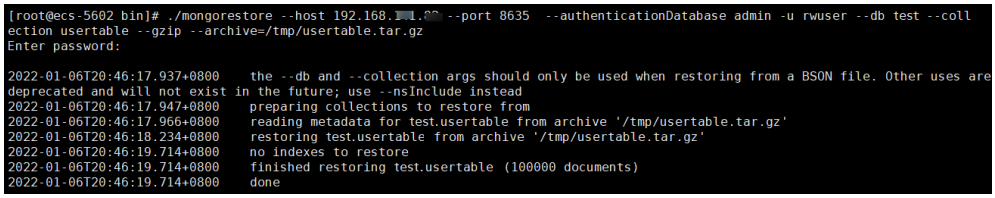
Parameter	Description
AUTH_DB	The database that authenticates *DB_USER*. Generally, the value is **admin**.
DB_USER	Account name of the database administrator. The default value is **rwuser**.
FILE_PATH	Path for storing the root certificate
DB_NAME	Name of the database to which data is to be imported
DB_COLLECTION	Collection in the database to be imported
Backup_directory	Directory for storing backup files

Enter the database administrator password when prompted:

```
Enter password:
```

Example:

./mongorestore --host 192.168.xx.xx --port 8635 --authenticationDatabase admin -u rwuser --db test --collection usertable --gzip --archive=/tmp/usertable.tar.gz



----End

Related Issues

When you back up the entire instance using mongodump and mongorestore, the permission verification fails.

- Cause

  The **rwuser** user has limited permissions on the **admin** and **config** databases of the instance. As a result, the permission verification fails.

- Solution

  Grant permissions on certain databases and tables to the user.

4.3 Instance Lifecycle

4.3.1 Restarting an Instance

Scenarios

You may need to occasionally restart a DB instance to perform routine maintenance.

Precautions

- If the instance status is **Available**, **Abnormal**, or **Checking restoration**, you can restart the instance.
- Restarting an instance will interrupt services, so exercise caution when performing this operation.
- If you restart an instance, all nodes in the instance are also restarted.
- If you enable operation protection to improve the security of your account and cloud products, two-factor authentication is required for sensitive operations. For details about how to enable operation protection, see ***Identity and Access Management User Guide***.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, locate the instance you want to restart and in the **Operation** column choose **Restart**.

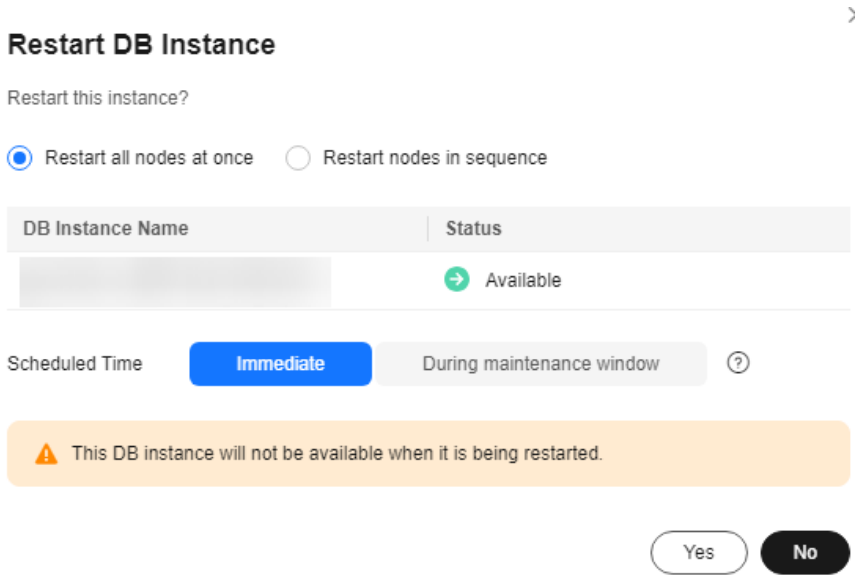
Alternatively, click the target DB instance, and on the displayed **Basic Information** page, click **Restart** in the upper right corner of the page.

Step 4 If you have enabled operation protection, click **Start Verification** in the **Restart DB Instance** dialog box. On the displayed page, click **Send Code**, enter the verification code, and click **Verify**. The page is closed automatically.

Step 5 In the displayed dialog box, click **Yes**.

For GeminiDB Mongo instances, you can restart several nodes at the same time or in sequence based on service requirements.

Figure 4-2 Restarting the GeminiDB Mongo instance



----End

4.3.2 Deleting a Pay-per-Use Instance

Scenarios

You can choose to delete a pay-per-use instance on the **Instances** page based on service requirements. To delete a yearly/monthly instance, unsubscribe from it. For details, see **Unsubscribing from a Yearly/Monthly DB Instance**.

Precautions

- Instances that an operation is being performed on cannot be deleted. They can be deleted only after the operations are complete.
- If a pay-per-use instance is deleted, its automated backups will also be deleted and you will no longer be billed for them. Manual backups, however, will be retained and generate additional costs.
- After an instance is deleted, all its data and all automated backups are automatically deleted as well and cannot be recovered. Back up it before you delete an instance. For details, see **Creating a Manual Backup**.
- After you delete an instance, all of its nodes are deleted.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, locate the target instance and in the **Operation** column choose **Delete** or **More** > **Delete**.

Step 4 If you have enabled operation protection, click **Start Verification** in the **Delete DB Instance** dialog box. On the displayed page, click **Send Code**, enter the verification code, and click **Verify**. The page is closed automatically.

NOTE

If you enable operation protection, two-factor authentication is required for sensitive operations to secure your account and cloud products. For details about how to enable operation protection, see ***Identity and Access Management User Guide***.

Step 5 In the displayed dialog box, click **Yes**.

Deleted DB instances are not displayed in the instance list.

----End

4.3.3 Recycling an Instance

GeminiDB Mongo API can move unsubscribed yearly/monthly instances and deleted pay-per-use DB instances to the recycle bin, so you can rebuild the DB instance and restore data from it.

Precautions

- If you delete an instance of full storage, the deleted instance will not be moved to the recycle bin.
- The recycling policy is enabled by default and cannot be disabled. Instances in the recycle bin are retained for 7 days by default, and this will not incur any charges.
- Currently, you can put up to 100 instances into the recycle bin. If the maximum number of instances is reached, you cannot put instances into the recycle bin anymore.
- You can modify the retention period, and the changes only apply to the DB instances deleted after the changes, so exercise caution when performing this operation.

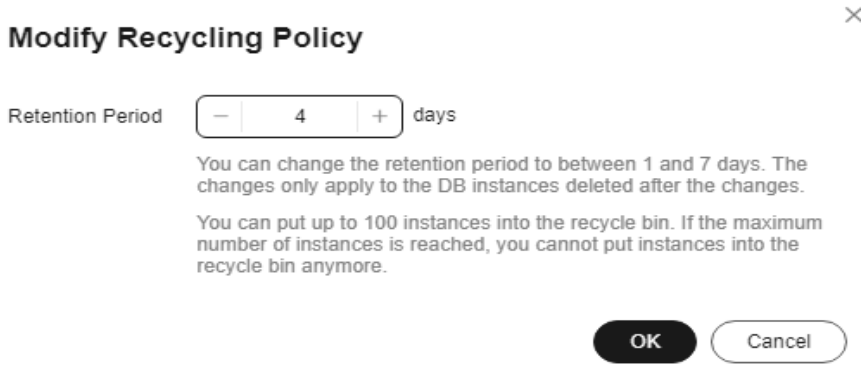
Modifying the Recycling Policy

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Recycling Bin** page, click **Modify Recycling Policy**. In the displayed dialog box, set the retention period from 1 day to 7 days. Then, click **OK**.

Figure 4-3 Modifying the recycling policy



----End

Rebuilding an Instance

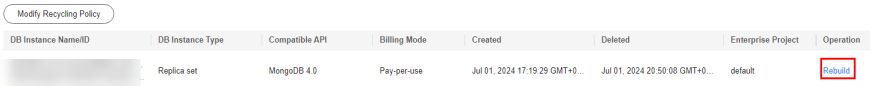
You can rebuild DB instances from the recycle bin within the retention period to restore data.

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Recycling Bin** page, locate the instance that you want to rebuild and click **Rebuild** in the **Operation** column.

Figure 4-4 Rebuilding an instance



Step 4 On the displayed page, set required parameters and submit the rebuilding task.

----End

4.4 Instance Modifications

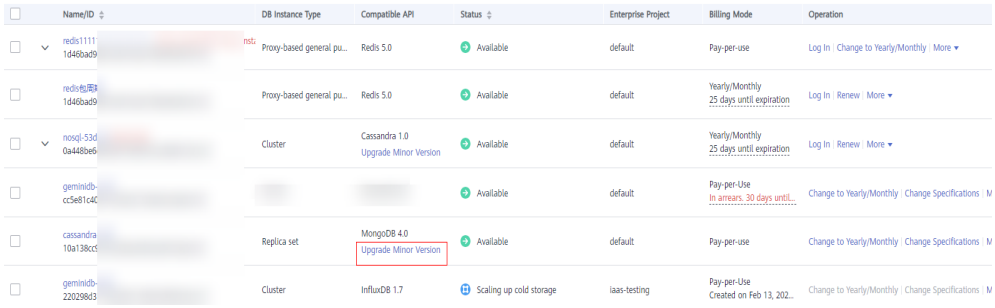
4.4.1 Upgrading Patches

GeminiDB Mongo can be upgraded by installing patches to improve performance, release new features, or fix bugs.

After a new patch version involving performance improvement, new functions, or problem rectification is released for GeminiDB Mongo, you can upgrade your instance to the latest version at a proper time based on service requirements.

If a new patch is released, you can upgrade your instance by clicking the upgrade button in the **Compatible API** column on the **Instances** page, as shown in **Figure 4-5**.

Figure 4-5 Installing a patch



Precautions

- Upgrade your instance once there are new patches released.
- If the database version is a risky version, the system prompts you to upgrade the database patch.
- Upgrading the minor version of an instance will restart each node of the instance in sequence. When a node is being restarted, its services will be taken over by another node. Each takeover will interrupt services for 3 to 5 seconds. So, perform an upgrade during off-peak hours and enable automatic reconnection so that each node can be reconnected immediately after being restarted.
- Upgrading basic components takes about 15 minutes. Upgrading data components takes about 1 to 2 minutes, which depends on how many nodes there are.
- The instance will be restarted and services may be interrupted during the upgrade. The interruption duration depends on services, quantity of nodes, and the amount of service data. Upgrade your instance during off-peak hours.

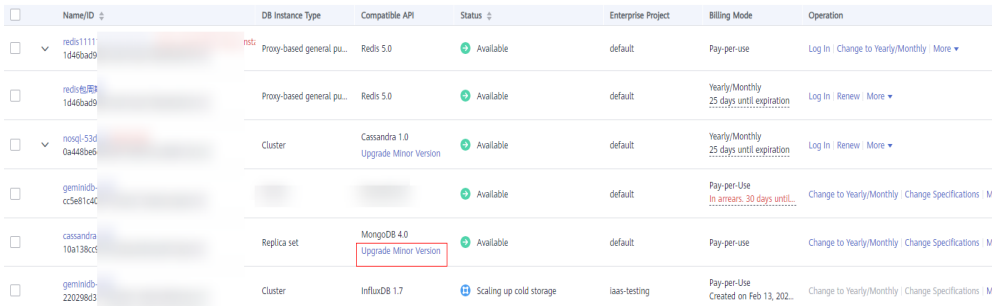
Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

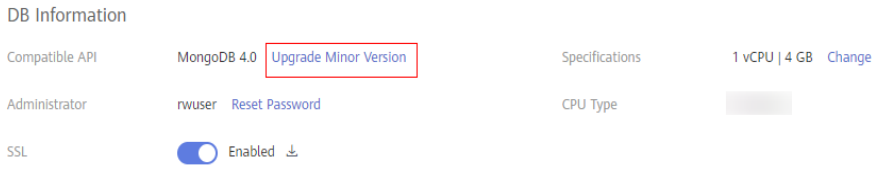
Step 3 On the **Instances** page, locate the instance you want to upgrade and click **Upgrade Minor Version** in the **Compatible API** column.

Figure 4-6 Installing a patch



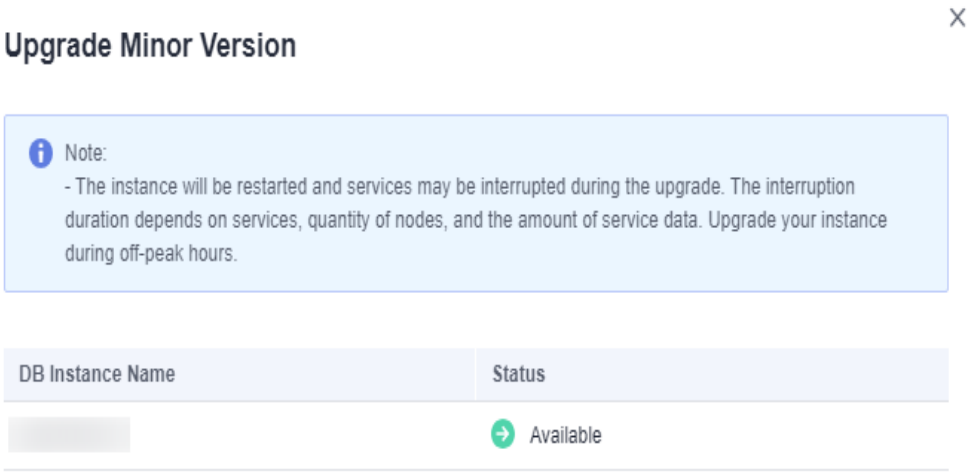
Alternatively, click the instance name to go to the **Basic Information** page. In the **DB Information** area, click **Upgrade Minor Version** in the **Compatible API** field.

Figure 4-7 Installing a patch



Step 4 In the displayed dialog box, click **OK**.

Figure 4-8 Confirming dialog box



Step 5 View the upgrade result on the **Instances** page.

- When the upgrade is ongoing, the instance status is **Upgrading minor version**.
- After the upgrade is complete, the instance status changes **Available**.

----End

4.4.2 Modifying the Name of an Instance

Scenarios

This section describes how to change an instance name to identify different instances.

Method 1

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, click  to the right of the instance whose name you want to modify.

- To submit the change, click **OK**.
- To cancel the change, click **Cancel**.

NOTE

The instance name:

- Can be the same as an existing instance name.
- Can include 4 to 64 bytes and must start with a letter. It is case-sensitive and allows only letters, digits, hyphens (-), and underscores (_).

Step 4 View the change results on the **Instances** page.

----End

Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose name you want to change and click its name.

Step 4 In the **Instance Information** area on the **Basic Information** page, click  in the **DB Instance Name** field to change the instance name.

- To submit the change, click  .
- To cancel the change, click  .

NOTE

The instance name:

- Can be the same as an existing instance name.
- Can include 4 to 64 bytes and must start with a letter. It is case-sensitive and allows only letters, digits, hyphens (-), and underscores (_).

Step 5 View the results on the **Instance Management** page.

----End

4.4.3 Resetting the Administrator Password

Scenarios

For security reasons, regularly change your administrator password.

Precautions

- If the instance status is **Available**, **Backing up**, **Checking restoration**, **Scaling up** or certain nodes become abnormal, you can reset the administrator password.

- If you enable operation protection to improve the security of your account and cloud products, two-factor authentication is required for sensitive operations. For details about how to enable operation protection, see ***Identity and Access Management User Guide***.

Method 1

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance you want to reset the password for and choose **More** > **Reset Password** in the **Operation** column.

Step 4 Enter and confirm the new administrator password and click **OK**.

The password must be 8 to 32 characters in length and contain uppercase letters, lowercase letters, digits, and any of the following special characters: ~!@#%^*-_=+?

Step 5 If you have enabled operation protection, click **Start Verification** in the displayed dialog box. On the displayed page, click **Send Code**, enter the verification code, and click **Verify**. The page is closed automatically.

----End

Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance. The **Basic Information** page is displayed.

Step 4 In the **DB Information** area, click **Reset Password** in the **Administrator** field.

Step 5 Enter and confirm the new administrator password and click **OK**.

The password must be 8 to 32 characters in length and contain uppercase letters, lowercase letters, digits, and any of the following special characters: ~!@#%^*-_=+?

Step 6 If you have enabled operation protection, click **Start Verification** in the displayed dialog box. On the displayed page, click **Send Code**, enter the verification code, and click **Verify**. The page is closed automatically.

----End

4.4.4 Scaling Up Storage Space

Scenarios

This section describes how to scale up the storage space of a DB instance to suit your service requirements.

During the scale-up process, the DB instance will not restart, and your services will not be interrupted.

Setting the Disk as Read-only When the Disk Space Is Full

To ensure that the GeminiDB Mongo instance can still be used if the storage space is about to be used up, the database is set as read-only, and data cannot be modified. If this happens, you can add more storage to restore the database to read/write status.

Table 4-7 Instance read-only upon full storage

Disk Space	Description
Less than 600 GB	• When the disk usage reaches 97%, the DB instance status is set to read-only. • When the disk usage decreases to 85%, the read-only state is automatically canceled for the DB instance.
Greater than or equal to 600 GB	• The remaining disk space is less than 18 GB, and the DB instance status is set to read-only. • If the remaining disk space is greater than or equal to 90 GB, the read-only state is automatically canceled for the DB instance.

Precautions

- Storage space can only be scaled up.
- To view storage details on the **Instances** page, contact customer service.

Method 1

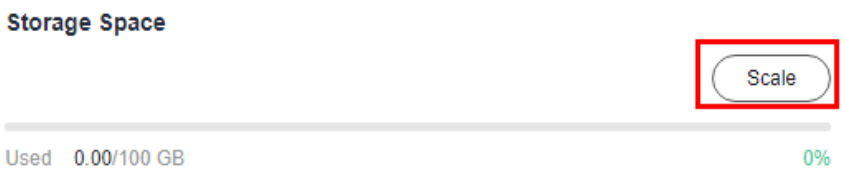
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance that you want to scale.

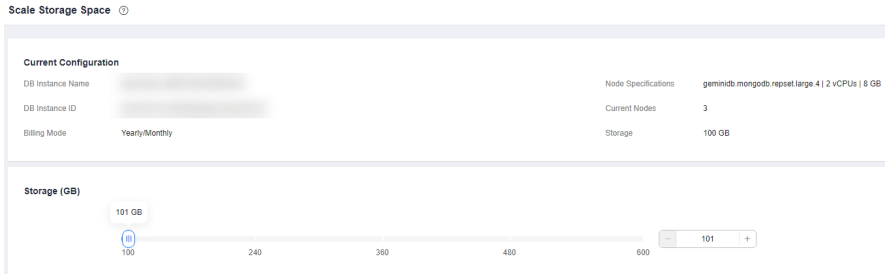
Step 4 In the **Storage Space** area on the **Basic Information** page, click **Scale**.

Figure 4-9 Scaling up storage space



Step 5 On the displayed page, specify the new storage capacity and click **Next**.

Figure 4-10 Scaling up storage space



Select at least 1 GB each time you scale up the storage, and the storage size must be an integer.

Step 6 On the displayed page, confirm the storage space.

- For yearly/monthly instance
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify the specifications, click **Submit** to scale up the storage space.

Step 7 Check the scaling-up result.

- The status of the DB instance in the instance list is **Scaling up**.
- After the scale up is completed, the instance status becomes **Available**.
- In the **Storage Space** area on the **Basic Information** page, check whether the scale up was successful.

----End

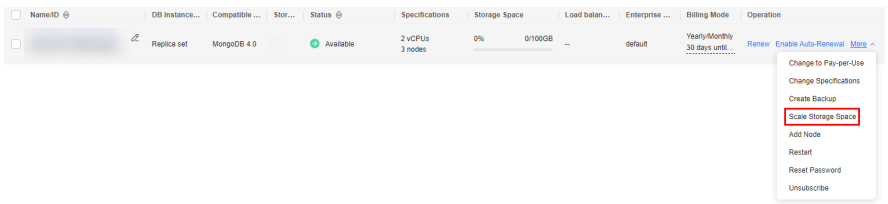
Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

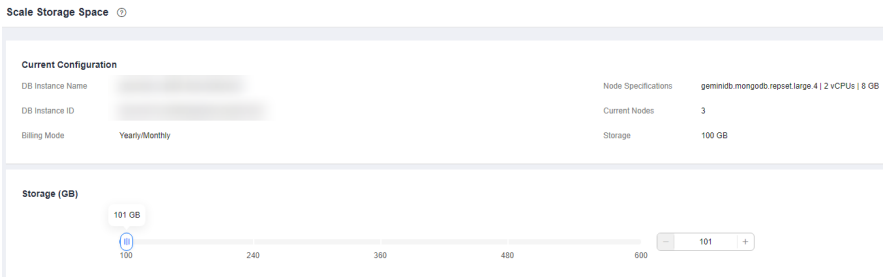
Step 3 On the **Instance Management** page, locate the instance whose storage space you want to scale up and choose **More** > **Scale Storage Space** in the **Operation** column.

Figure 4-11 Scaling up storage space



Step 4 On the displayed page, specify the new storage capacity and click **Next**.

Figure 4-12 Scaling up storage space



Select at least 1 GB each time you scale up the storage, and the storage size must be an integer.

Step 5 On the displayed page, confirm the node configuration details.

- For yearly/monthly DB instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Next** and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to add the nodes.

Step 6 Check the scaling-up result.

- The status of the DB instance in the instance list is **Scaling up**.
- After the scale up is completed, the instance status becomes **Available**.
- In the **Storage Space** area on the **Basic Information** page, check whether the scale up was successful.

----End

4.4.5 Changing Specifications of an Instance

Scenarios

This section describes how to change your instance specifications to suit your service requirements.

Precautions

- Instances can be scaled up or down by changing their specifications.
- If one instance has multiple nodes, the change will be performed on the nodes one by one. It takes about 5 to 10 minutes for each node, and the total time required depends on the number of the nodes.

- For a node whose specifications are being changed, its computing tasks are handed over to other nodes. Change specifications of nodes during off-peak hours to prevent the instance from overload.
- To view vCPUs and nodes on the **Instances** page, contact customer service.
- If you forcibly change the class of an abnormal node in a DB instance, services may be interrupted.

  NOTE

  To forcibly change the class of an abnormal node, submit a service ticket by choosing **Service Tickets** > **Create Service Ticket** in the upper right corner of the management console.

Method 1

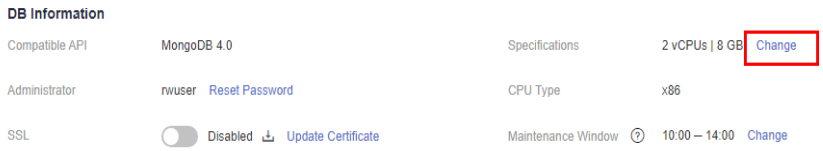
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose specifications you want to change and click its name.

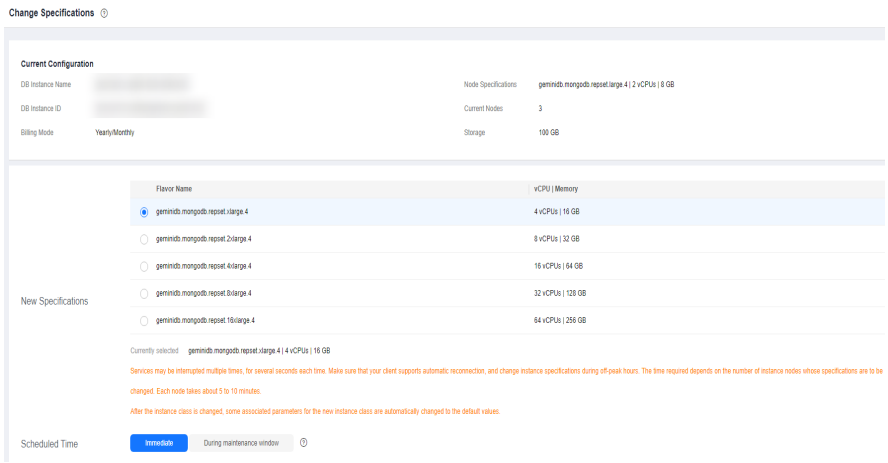
Step 4 In the **DB Information** area, click **Change** in the **Specifications** field.

Figure 4-13 Changing specifications



Step 5 On the displayed page, select the required specifications and click **Next**.

Figure 4-14 Changing specifications



Step 6 On the displayed page, confirm the specifications.

- For yearly/monthly instances

  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to change the specifications. If you are scaling up the specifications, go to the payment page, select a payment method, and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to change the specifications.

Step 7 View the specifications change result.

Go to the **Basic Information** page and in the **DB Information** area you can see the new specifications.

----End

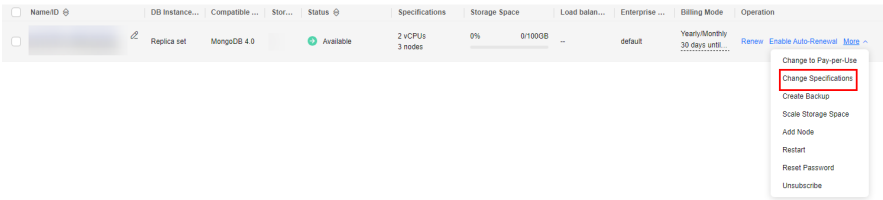
Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

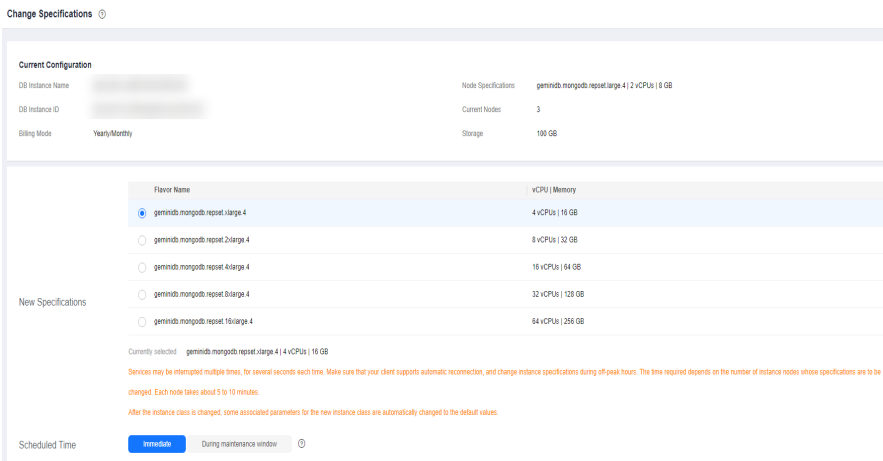
Step 3 On the **Instances** page, locate the instance whose specifications you want to change and choose **More** > **Change Specifications** in the **Operation** column.

Figure 4-15 Changing specifications



Step 4 On the displayed page, select the required specifications and click **Next**.

Figure 4-16 Changing specifications



Step 5 On the displayed page, confirm the specifications.

- For yearly/monthly instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to change the specifications. If you are scaling up the specifications, go to the payment page, select a payment method, and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to change the specifications.

Step 6 View the specifications change result.

Go to the **Basic Information** page and in the **DB Information** area you can see the new specifications.

----End

4.4.6 Changing a Maintenance Window

The default maintenance window is 10:00–14:00 (GMT+08:00) but you can change it if needed. To prevent service interruption, set the maintenance window to off-peak hours. Before calling this API:

Precautions

- This function for setting an maintenance window is still in the open beta test (OBT) phase. To use it, contact customer service.
- You can configure a maintenance window only for restarting a DB instance, changing an instance class, or upgrading the minor version of a DB instance.
- The specification change and patch upgrade that have been performed during the maintenance period cannot be performed immediately. The instance can be restarted immediately.
- You can cancel a task to be executed.
- Changing the maintenance window will not affect the timing that has already been scheduled.
- The maintenance window cannot overlap the time window configured for backups. Otherwise, scheduled tasks may fail.
- During the maintenance window, the scheduled task is scanned and executed every 10 minutes. If the task is delivered near the end of the maintenance period, the task may fail to be scanned and the execution is canceled.

Changing a Maintenance Window

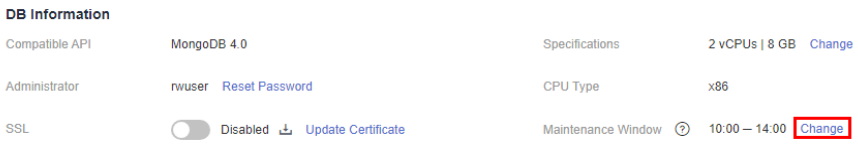
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance whose specifications you want to change. The **Basic Information** page is displayed.

Step 4 On the **Bacis Information** page, locate **Maintenance Window** and click **Change**.

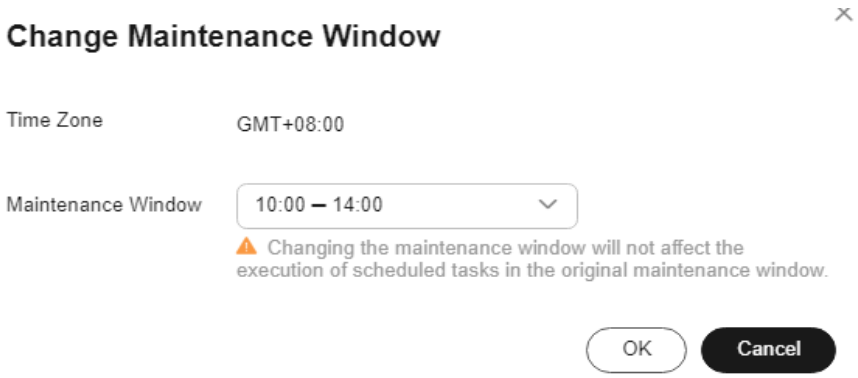
Figure 4-17 The change button



Step 5 On the **Change Maintainable Window** page, select the maintenance time period as needed, and then click **OK**.

Supported time periods: 02:00-06:00, 06:00-10:00, 10:00-14:00, 14:00-18:00, 18:00-22:00, and 22:00-02:00

Figure 4-18 Changing a maintenance window



Step 6 Check the result.

On the **Basic Information** page, you can view the changed maintenance window.

----End

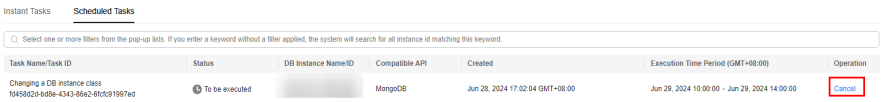
Canceling a Scheduled Task

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Task Center** page, locate a scheduled task, and click **Cancel** in the **Operation** column.

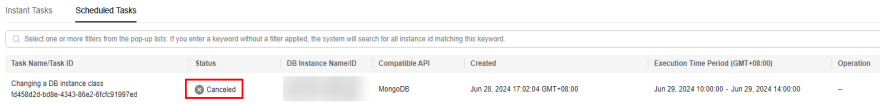
Figure 4-19 Canceling a task



Step 4 Check the result.

On the **Task Center** page, you can view the result. After the task is cancelled, its status changes to **Cancelled**.

Figure 4-20 Checking cancelled tasks



----End

4.4.7 Adding Nodes

Scenarios

This section describes how to add nodes to a DB instance to suit your service requirements. You can also delete a node as required. For details, see **Deleting Nodes**.

Precautions

- GeminiDB Mongo allows you to add nodes only to 4.0 replica set instances. The role of each new node is Readonly.
- Adding nodes may lead to the decrease of OPS. You are advised to perform this operation during off-peak hours.
- You can only add nodes when the instance status is **Available** or **Checking restoration**.
- An instance cannot be deleted when one or more nodes are being added.
- If the storage is insufficient, adding nodes is not supported. Expand the storage first. For details about the storage supported by instances of different specifications, see **Instance Specifications**.

Method 1

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

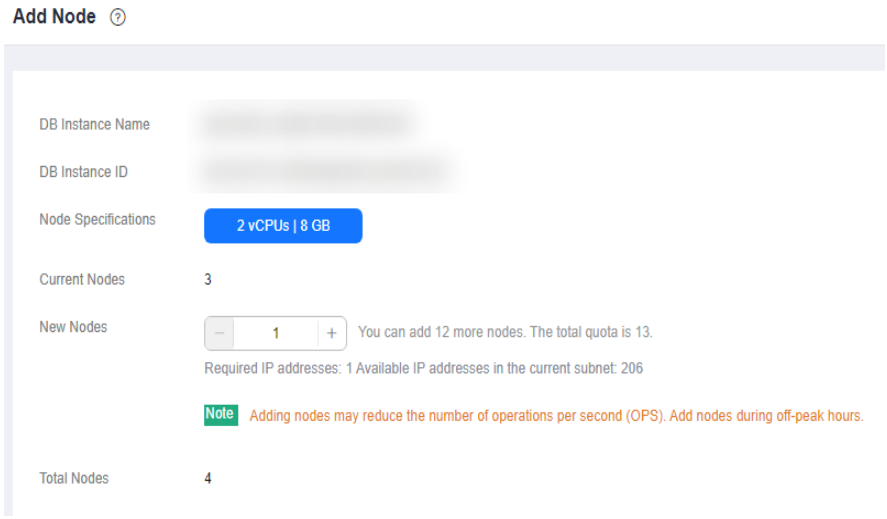
Step 3 On the **Instances** page, click the target DB instance.

Step 4 In the **Node Information** area on the **Basic Information** page, click **Add Node**.

Figure 4-21 Node information

Step 5 Specify **Add Nodes** and click **Next**.

Figure 4-22 Add Node



New nodes are of the same specifications as existing nodes. Once a new node is added, its specifications cannot be changed.

Step 6 On the displayed page, confirm the node configuration details.

- For yearly/monthly DB instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Next** and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to add the nodes.

Step 7 View the result of adding nodes.

- The status of the DB instance in the instance list is **Adding node**.
- After the nodes are added, the DB instance status becomes **Available**.
- Click the DB instance name. In the **Node Information** area on the **Basic Information** page, view the information about the new nodes.

----End

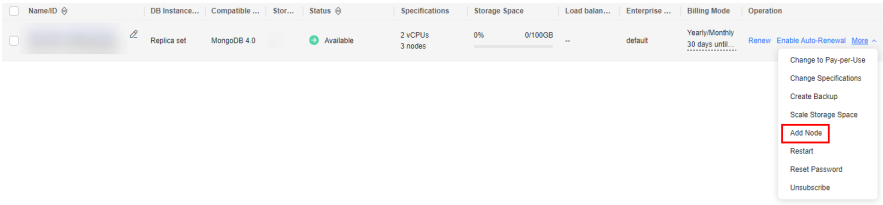
Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

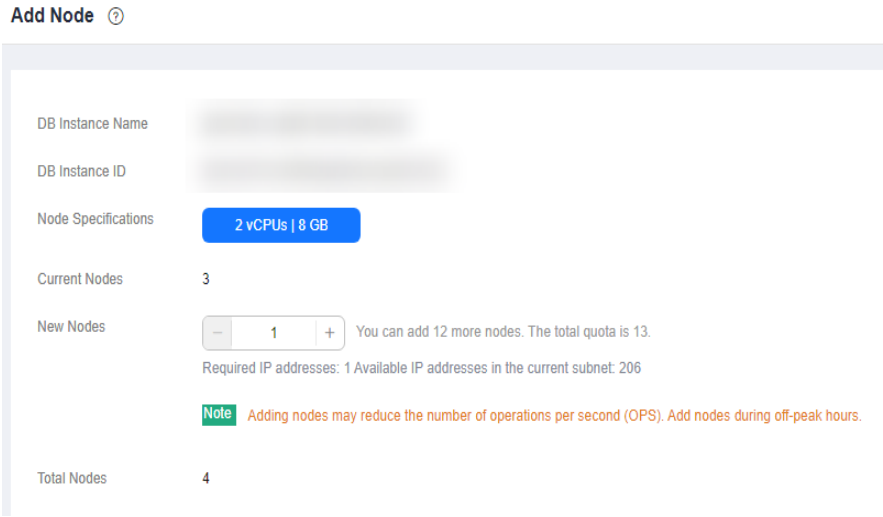
Step 3 On the **Instance Management** page, locate the instance you want to add nodes for and choose **More** > **Add Node** in the **Operation** column.

Figure 4-23 Adding nodes



Step 4 Specify **Add Nodes** and click **Next**.

Figure 4-24 Add Node



New nodes are of the same specifications as existing nodes. Once a new node is added, its specifications cannot be changed.

Step 5 On the displayed page, confirm the node configuration details.

- For yearly/monthly DB instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Next** and complete the payment.
- For pay-per-use instances
  - If you need to modify your settings, click **Previous** to go back to the page where you specify details.
  - If you do not need to modify your settings, click **Submit** to add the nodes.

Step 6 View the result of adding nodes.

- The status of the DB instance in the instance list is **Adding node**.
- After the nodes are added, the DB instance status becomes **Available**.

- Click the DB instance name. In the **Node Information** area on the **Basic Information** page, view the information about the new nodes.

----End

4.4.8 Deleting Nodes

Scenarios

You can delete nodes that are no longer used to release resources.

Precautions

- GeminiDB Mongo allows you to delete nodes only from 4.0 replica set instances. The role of each node to be deleted is Readonly.
- Nodes cannot be deleted from yearly/monthly DB instances.
- Deleted nodes cannot be recovered. Exercise caution when performing this operation.
- If you enable operation protection to improve the security of your account and cloud products, two-factor authentication is required for sensitive operations. For details about how to enable operation protection, see ***Identity and Access Management User Guide***.

Procedure

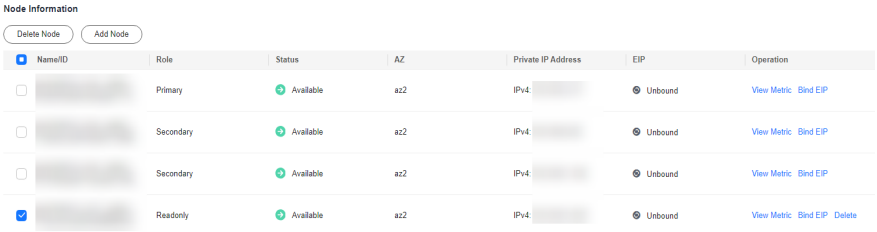
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the target DB instance.

Step 4 In the **Node Information** area on the **Basic Information** page, locate the target node and click **Delete**.

Figure 4-25 Node information



Step 5 If you have enabled operation protection, click **Start Verification** in the **Delete Node** dialog box. On the displayed page, click **Send Code**, enter the verification code, and click **Verify**. The page is closed automatically.

Step 6 In the displayed dialog box, click **Yes**.

- The status of the DB instance in the instance list is **Deleting node**.
- After the deletion, the DB instance status becomes **Available**.

----End

4.4.9 Managing Tags

Scenarios

Tag Management Service (TMS) enables you to use tags on the management console to manage resources. TMS works with other cloud services to manage tags. TMS manages tags globally and other cloud services manage their own tags.

Adding tags to GeminiDB Mongo resources helps you better identify and manage them. A DB instance can be tagged during or after it is created.

After a DB instance is tagged, you can search for the tag key or value to quickly query the instance details.

Precautions

- You are advised to set predefined tags on the TMS console.
- A tag consists of a key and value. You can add only one value for each key. For details about the naming rules of tag keys and tag values, see **Table 4-8**.
- By default, each instance supports up to 10 tags. To use more tags, contact customer service to apply for a quota of 20 tags.
- The tag name must comply with the naming rules described in **Table 4-8**.

Table 4-8 Naming rules

Parameter	Requirement	Example Value
Tag key	• Cannot be left blank. • Must be unique for each instance. • Contains a maximum of 36 characters. • Can only consist of digits, letters, underscores (_), and hyphens (-).	Organization
Tag value	• Can be left blank. • Contains a maximum of 43 characters. • Can only consist of digits, letters, underscores (_), periods (.), and hyphens (-).	nosql_01

Adding a Tag

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance. The **Basic Information** page is displayed.

Step 4 In the navigation pane on the left, choose **Tags**.

Step 5 On the **Tags** page, click **Add Tag**. In the displayed dialog box, enter a tag key and value, and click **OK**.

Step 6 View and manage the tag on the **Tags** page.

----End

Editing a Tag

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance. The **Basic Information** page is displayed.

Step 4 In the navigation pane on the left, choose **Tags**.

Step 5 On the **Tags** page, locate the tag to be edited and click **Edit** in the **Operation** column. In the displayed dialog box, change the tag value and click **OK**.

Only the tag value can be edited.

Step 6 View and manage the tag on the **Tags** page.

----End

Deleting a Tag

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance. The **Basic Information** page is displayed.

Step 4 In the navigation pane on the left, choose **Tags**.

Step 5 On the **Tags** page, locate the tag to be deleted and click **Delete** in the **Operation** column. In the displayed dialog box, click **Yes**.

Step 6 View that the tag is no longer displayed on the **Tags** page.

----End

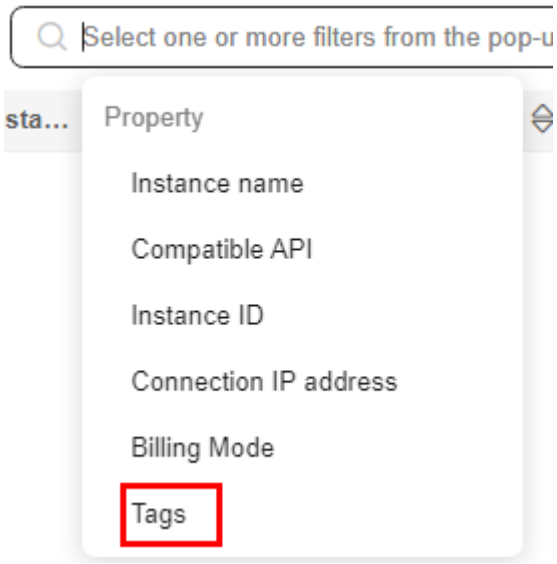
Search by Tag

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

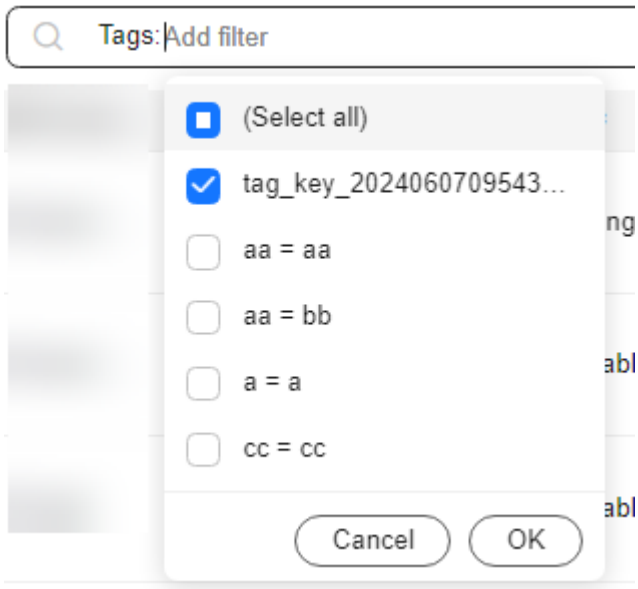
Step 3 On the **Instances** page, select **Tags** in the search box.

Figure 4-26 Selecting tags



Step 4 Select the tag to be queried and click **OK** to query information about instances associated with the tag.

Figure 4-27 Searching by tag



----End

4.4.10 Updating the OS of an Instance

To improve database performance and security, the OS of a GeminiDB Mongo instance needs to be updated in a timely manner.

Every time you upgrade the kernel version of your instance, GeminiDB Mongo determines whether to update the OS and selects the right cold patch to upgrade the OS if necessary.

Updating the OS does not change the DB instance version or other information.

In addition, GeminiDB Mongo installs hot patches as required to fix major OS vulnerabilities within the maintenance window you specified.

4.5 Connections

4.5.1 Configuring Security Group Rules

A security group is a collection of access control rules for ECSs and GeminiDB Mongo instances that have the same security protection requirements and are mutually trusted in a VPC.

This section describes how to create a security group to enable specific IP addresses and ports to access GeminiDB Mongo API.

This section describes how to configure security group rules when you connect to a GeminiDB Mongo instance through a private or public network.

Precautions

- By default, you can create up to 500 security group rules.
- Too many security group rules will increase the first packet latency, so a maximum of 50 rules for each security group is recommended.
- One security group can be associated with only one GeminiDB Mongo instance.
- For details about how to configure security group rules, see **Table 4-9**.

Table 4-9 Parameter description

Scenario	Description
Connecting to an instance over a private network	Check whether the ECS and GeminiDB Mongo instance are in the same security group. • If the ECS and GeminiDB Mongo instance are in the same security group, they can communicate with each other by default. No security group rule needs to be configured. • If the ECS and GeminiDB Mongo instance are in different security groups, configure security group rules for the ECS and DB instance, respectively. – GeminiDB Mongo instance: Configure inbound rules for the security group associated with the instance. For details, see **Procedure**. – ECS: The default security group rule allows all outbound data packets, so you do not need to configure a security rule for the ECS. If not all outbound traffic is allowed in the security group, configure an outbound rule for the ECS.

Scenario	Description
Connecting to an instance over a public network	To access a GeminiDB Mongo instance over a public network, add an inbound rule for the security group associated with the instance. For details, see **Procedure**.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance.

Step 4 Configure security group rules.

Method 1

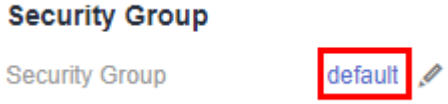
In the **Network Information** area on the **Basic Information** page, click the security group.

Figure 4-28 Security Group

Method 2

On the **Basic Information** page, choose **Connections** in the navigation pane on the left. In the **Security Group** area on the right, click the name of the security group. The **Security Group** page is displayed.

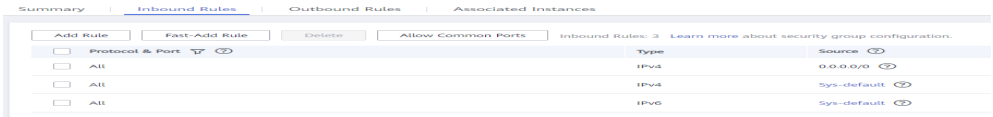
Figure 4-29 Security Group



Step 5 Add an inbound rule.

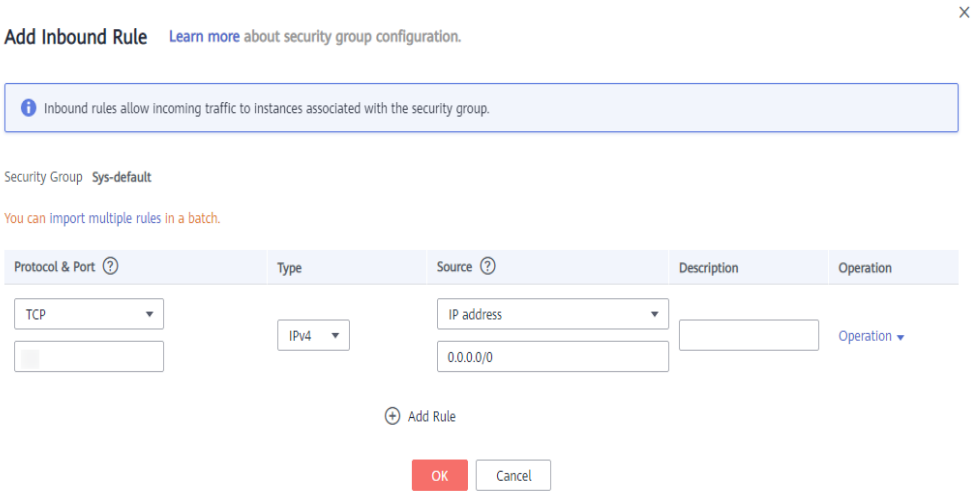
1. Click the **Inbound Rules** tab.

Figure 4-30 Inbound rules



2. Click **Add Rule**. The **Add Inbound Rule** dialog box is displayed.

Figure 4-31 Adding a rule



3. In the displayed dialog box, set required parameters.

Table 4-10 Inbound rule settings

Parameter	Description	Example Value
Protocol & Port	– The network protocol required for access. Available options: **All**, **TCP**, **UDP**, **ICMP**, or **GRE** – **Port**: The port (1 to 65535) for accessing the ECS.	TCP
Type	IP address type. This parameter is available after IPv6 is enabled. – IPv4 – IPv6	IPv4
Source	The IP address, IP address group, or security group that the rule applies to, which allows access from IP addresses or instances in other security group. Example: – Single IP address: xxx.xxx.xxx.xxx/32 (IPv4) – Subnet: xxx.xxx.xxx.0/24 – All IP addresses: 0.0.0.0/0 – sg-abc (security group)	0.0.0.0/0
Description	(Optional) Provides supplementary information about the security group rule. The description can contain up to 255 characters and cannot contain angle brackets (<>).	-

Step 6 Click **OK**.

----End

4.5.2 Binding and Unbinding an EIP

Scenarios

After you create a GeminiDB Mongo instance, you can bind an EIP to it to allow external access. If later you want to prohibit external access, you can also unbind the EIP from the instance.

Precautions

- Before accessing a database, apply for an EIP on the VPC console. Then, add an inbound rule to allow the IP addresses or IP address ranges of ECSs. For details, see **Configuring Security Group Rules**.
- To change the EIP that has been bound to a node, unbind it from the node first.

Binding an EIP

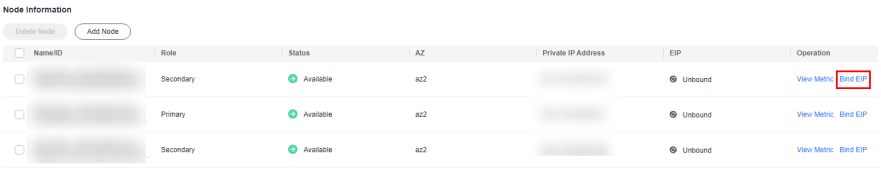
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance that you want to bind an EIP to and click its name.

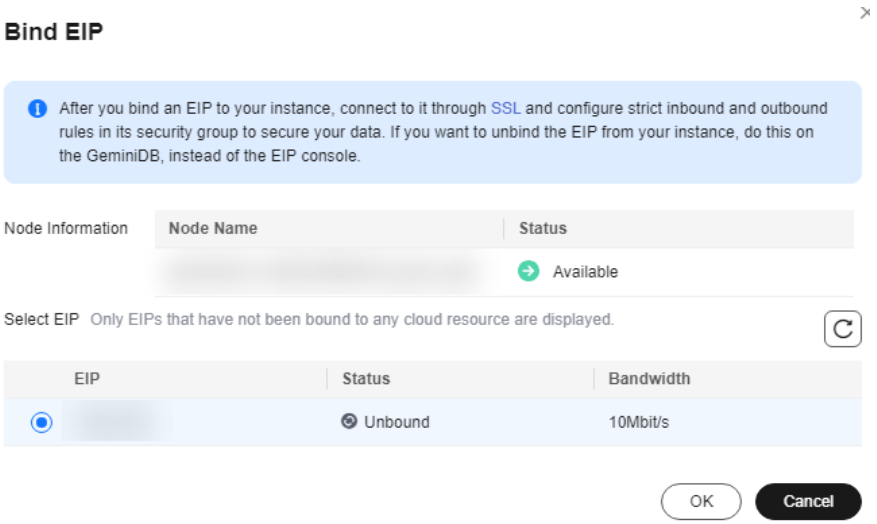
Step 4 On the **Basic Information** page, in the **Node Information** area, locate the target node and click **Bind EIP** in the **Operation** column.

Figure 4-32 Binding an EIP



Step 5 In the displayed dialog box, all available unbound EIPs are listed. Select the required EIP and click **Yes**. If no available EIPs are displayed, click **View EIP** and create an EIP on the VPC console.

Figure 4-33 Selecting an EIP



Step 6 In the **EIP** column, view the EIP that is successfully bound.

To unbind the EIP from the DB instance, see **Unbinding an EIP**.

----End

Unbinding an EIP

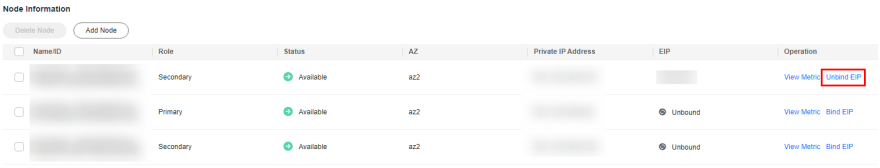
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, click the instance that you want to unbind an EIP from.

Step 4 On the **Basic Information** page, in the **Node Information** area, locate the target node and click **Unbind EIP** in the **Operation** column.

Figure 4-34 Unbinding an EIP



Step 5 In the displayed dialog box, click **Yes**.

To bind an EIP to the DB instance again, see **Binding an EIP**.

----End

4.5.3 Configuring an SSL Connection

Scenarios

Secure Socket Layer (SSL) is an encryption-based Internet security protocol for establishing an encrypted link between a server and a client. It provides privacy, authentication, and integrity to Internet communications.

- Authenticates users and servers, ensuring that data is sent to the correct clients and servers.
- Encrypts data to prevent it from being intercepted during transfer.
- Ensures data integrity during transmission.

After SSL is enabled, you can establish an encrypted connection between your client and the instance you want to access to improve data security.

Precautions

- To use SSL, contact customer service to apply for the required permissions.
- Enabling or disabling SSL will cause instances to restart. Exercise caution when performing this operation.
- If SSL is enabled, you can connect to a database using SSL to improve security.

  Encryption algorithms that may have security risks are not allowed. Secure encryption algorithms and supported cipher suits are described **Table 4-11**.

  Table 4-11 Secure encryption algorithms and supported cipher suits

Version	TLS Version	Cipher Suite
4.0	TLS 1.2	DHE-RSA-AES256-GCM-SHA384 DHE-RSA-AES128-GCM-SHA256

  The server where the client is located must support the corresponding TLS version and encryption algorithm suite. Otherwise, the connection fails.
- If SSL is disabled, you can connect to a database using an unencrypted connection.

Enabling SSL

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance. The **Basic Information** page is displayed.

Step 4 In the **DB Information** area, click  to enable the SSL toggle.

Alternatively, choose **Connections** in the navigation pane on the left. On the **Basic Information** page, click  to enable the SSL toggle.

Step 5 In the displayed dialog box, click **Yes**.

Step 6 In the **Basic Information** area, view the results.

Step 7 After SSL is enabled, click next to **SSL** to download an SSL certificate.

For details about how to connect to an instance using an SSL connection, see **SSL Connection**.

----End

Disabling SSL

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, click the instance.

Step 4 In the **DB Information** area on the **Basic Information** page, click next to the **SSL** field.

Alternatively, in the navigation pane on the left, choose **Connections**. In the **Basic Information** area, click next to the **SSL** field.

Step 5 In the displayed dialog box, click **Yes**.

Step 6 In the **Basic Information** area, view the results.

Step 7 After SSL is disabled, you can connect to an instance using an unencrypted connection.

For details, see **Non-SSL Connection**.

----End

4.5.4 Changing a Database Port

Scenarios

This section describes how to modify the database port to ensure system security.

The database port cannot be changed when the instance is in any of the following statuses:

- Frozen
- Restarting
- Adding nodes
- Changing instance class
- Scaling up
- Deleting node

Precautions

This function is in the open beta test (OBT) phase. Contact customer service to apply for the function.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the target DB instance.

Step 4 In the navigation pane on the left, choose **Connections**.

Step 5 In the **Basic Information** area, click to right of the **Database Port** field.

The database port ranges from 2100 to 9500 and cannot be 8636, 8637, or 8638.

- To submit the change, click . This process takes about 1 to 5 minutes.
- To cancel the change, click .

Step 6 View the modification result.

----End

4.5.5 Changing a Security Group

Scenarios

You can change the security group for GeminiDB Mongo instances.

Precautions

- If you are adding nodes to an instance, the security group of the instance cannot be changed.
- This function is in the open beta test (OBT) phase. To use this function, contact customer service.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the target DB instance.

Step 4 In the navigation pane on the left, choose **Connections**.

Step 5 In the **Security Group** area, click beside the security group name and select the required security group.

- To submit the change, click . This process takes about 1 to 3 minutes.
- To cancel the change, click .

Step 6 View the modification result.

----End

4.5.6 Configuring Access Across CIDR Blocks

Access across CIDR blocks is required only when the CIDR blocks of the client and the replica set instance are different. For example, if the client CIDR block is

192.168.0.0/16 and the replica set instance's CIDR block is 172.16.0.0/24, add the CIDR block 192.168.0.0/16 so that the client can access the replica set instance.

This section describes how to configure access to an instance across CIDR blocks.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance you want to configure access across CIDR blocks for.

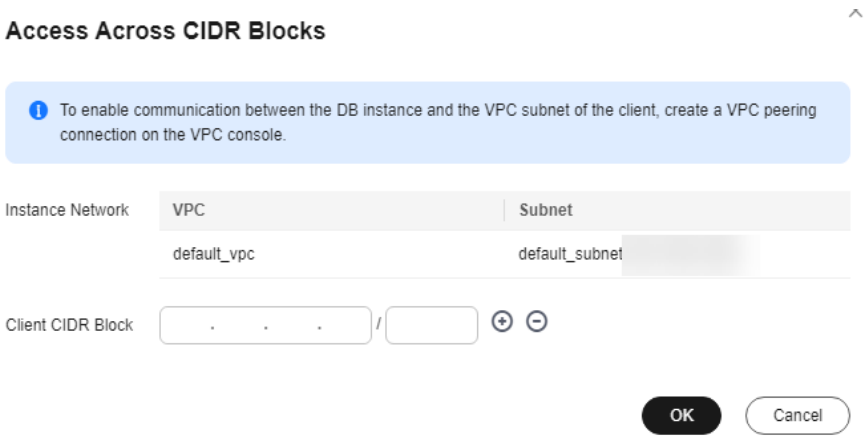
Step 4 In the navigation pane on the left, choose **Connections**.

Step 5 In the **Address** area, click **Enable** to the right of **Access Across CIDR Blocks** field.

Figure 4-35 Enabling access across CIDR blocks

Step 6 Configure the source client CIDR.

Figure 4-36 Access across CIDR blocks



- If there are multiple CIDR blocks, click ⊕ to add them in sequence. Up to 9 CIDR blocks can be configured, but you cannot insert duplicate blocks.
- The blocks starting with 127 are not allowed. The subnet mask ranges from 8 to 32.

Step 7 Confirm that access across CIDR blocks is enabled. **Enabled** is displayed to the right of the **Access Across CIDR Blocks** field.

To change the CIDR, click **Change** to the right of **Access Across CIDR Blocks** to add, modify, or delete CIDR blocks.

Figure 4-37 Modifying access across CIDR blocks

NOTE

To ensure the ECS and the DB instance can communicate with each other, configure the connection by referring to **VPC Peering Connection Overview**.

----End

4.6 Database Commands

4.6.1 Supported and Restricted Commands

After you connect to the GeminiDB Mongo database, pay attention to the following supported and restricted commands:

For more information, see **official MongoDB documentation**.

Table 4-12 Commands supported and restricted by version 4.0

Type	Command	Supported In Read/Write Mode	Remarks
Aggregates Commands	aggregate	√	-
	count	√	-
	distinct	√	-
	group	√	-
	mapReduce	√	-
Geospatial Commands	geoNear	√	-
	geoSearch	×	Not supported by replica sets.
Query and Write Operation Commands	find	√	-
	insert	√	-
	update	√	-
	delete	√	-
	findAndModify	√	-
	getMore	√	-

Type	Command	Supported In Read/Write Mode	Remarks
	getLastError	√	-
	parallelCollection-Scan	x	System command
Query Plan Cache Commands	planCacheListFil-ters	√	-
	planCacheSetFil-ter	√	-
	planCacheClearFil ters	√	-
	planCacheListQue ryShapes	√	-
	planCacheListPlan s	√	-
	planCacheClear	√	-
Authentication Commands	logout	√	-
	authenticate	√	-
	getnonce	√	-
User Management Commands	createUser	√	-
	updateUser	√	-
	dropUser	√	-
	dropAllUsersFrom Database	√	-
	grantRolesToUser	√	-
	revokeRolesFrom User	√	-
	usersInfo	√	-
Role Management Commands	invalidateUserC-ache	√	-
	createRole	√	-
	updateRole	√	-
	dropRole	√	-
	dropAllRolesFrom Database	√	-

Type	Command	Supported In Read/Write Mode	Remarks
	grantPrivileges-ToRole	√	-
	revokePrivileges-FromRole	√	-
	grantRolesToRole	√	-
	revokeRolesFrom-Role	√	-
	rolesInfo	√	-
Replication Commands	replSetFreeze	x	System command
	replSetGetStatus	√	Supported by replica sets.
	replSetInitiate	x	System command
	replSetMaintenance	x	System command
	replSetReconfig	x	System command
	replSetStepDown	x	System command
	replSetSyncFrom	x	System command
	resync	x	System command
	applyOps	x	System command
	isMaster	√	-
	replSetGetConfig	x	System command
Sharding Commands	flushRouterConfig	x	System command
	addShard	x	System command
	addShardToZone	x	System command
	balancerStart	x	System command
	balancerStatus	√	System command
	balancerStop	x	System command
	removeShardFrom Zone	x	System command
	updateZoneKeyRange	x	System command

Type	Command	Supported In Read/Write Mode	Remarks
	cleanupOrphaned	x	High-risk commands
	checkShardingIn-dex	x	System command
	enableSharding	√	-
	listShards	√	System command
	removeShard	x	High-risk commands
	getShardMap	x	System command
	getShardVersion	x	System command
	mergeChunks	x	System command
	setShardVersion	x	System command
	shardCollection	√	Currently, this command applies to empty tables only.
	shardingState	x	System command
	unsetSharding	x	System command
	split	x	System command
	splitChunk	x	System command
	splitVector	x	System command
	moveChunk	x	System command
	movePrimary	x	System command
	isdbgrid	√	-
Sessions Commands	abortTransaction	√	Supported by replica sets.
	commitTransactio n	√	Supported by replica sets.
	endSessions	√	-
	killAllSessions	√	-
	killAllSessionsBy-Pattern	√	-

Type	Command	Supported In Read/Write Mode	Remarks
	killSessions	√	-
	refreshSessions	√	-
	startSession	√	-
Administration Commands	setFeatureCompatibilityVersion	√	-
	renameCollection	√	-
	dropDatabase	√	-
	listCollections	√	-
	drop	√	-
	create	√	-
	clone	x	System command
	cloneCollection	x	System command
	cloneCollectionAsCapped	x	System command
	convertToCapped	√	-
	filemd5	√	-
	createIndexes	√	-
	listIndexes	√	-
	dropIndexes	√	-
	fsync	√	-
	clean	x	System command
	connPoolSync	x	System command
	connectionStatus	√	-
	compact	x	High-risk commands
	collMod	√	-
	reIndex	√	-
	setParameter	x	System configuration command
	getParameter	√	-

Type	Command	Supported In Read/Write Mode	Remarks
	repairDatabase	x	High-risk commands
	touch	x	The storage engine type is not supported.
	shutdown	x	High-risk commands
	logRotate	x	High-risk commands
	killOp	√	-
Diagnostic Commands	availableQueryOptions	√	-
	buildInfo	√	-
	collStats	√	-
	connPoolStats	x	System command
	dataSize	√	-
	dbHash	x	System command
	dbStats	√	-
	driverOIDTest	x	System command
	explain	√	-
	features	√	-
	getCmdLineOpts	x	System command
	getLog	x	System command
	hostInfo	x	System command
	isSelf	x	System command
	listCommands	√	-
	listDatabases	√	-
	netstat	x	System command
	ping	√	-
	profile	√	-
	serverStatus	√	-

Type	Command	Supported In Read/Write Mode	Remarks
	shardConnPoolStats	x	System command
	top	x	System command
	validate	x	System configuration command
	whatsmyuri	√	-
System Events Auditing Commands	logApplicationMessage	x	System command

4.7 Data Backup

4.7.1 Overview

GeminiDB Mongo API allows you to create backups for your instance to ensure data reliability. After an instance is deleted, the manual backup data is retained. Automated backup data is released together with instances. Backup data cannot be downloaded or exported.

Backup Methods

GeminiDB Mongo instances support automated and manual backups.

- Automated backup

  You can click **Modify Backup Policy** on the GeminiDB console, and the system will automatically back up your instance data based on the time window and backup cycle you set in the backup policy and will store the data for the retention period you specified.

  Automated backups cannot be manually deleted. You can adjust their retention period by referring to **Modifying an Automated Backup Policy** , and backups that expire will be automatically deleted.

- Manual backup

  A manual backup is a full backup of a DB instance and can be retained until you manually delete it. You can create a manual backup for your instance at any time to meet service requirements.

  Regularly backing up your database is recommended. If your database becomes faulty or data is corrupted, you can restore it from backups.

Table 4-13 Backup methods and scenarios

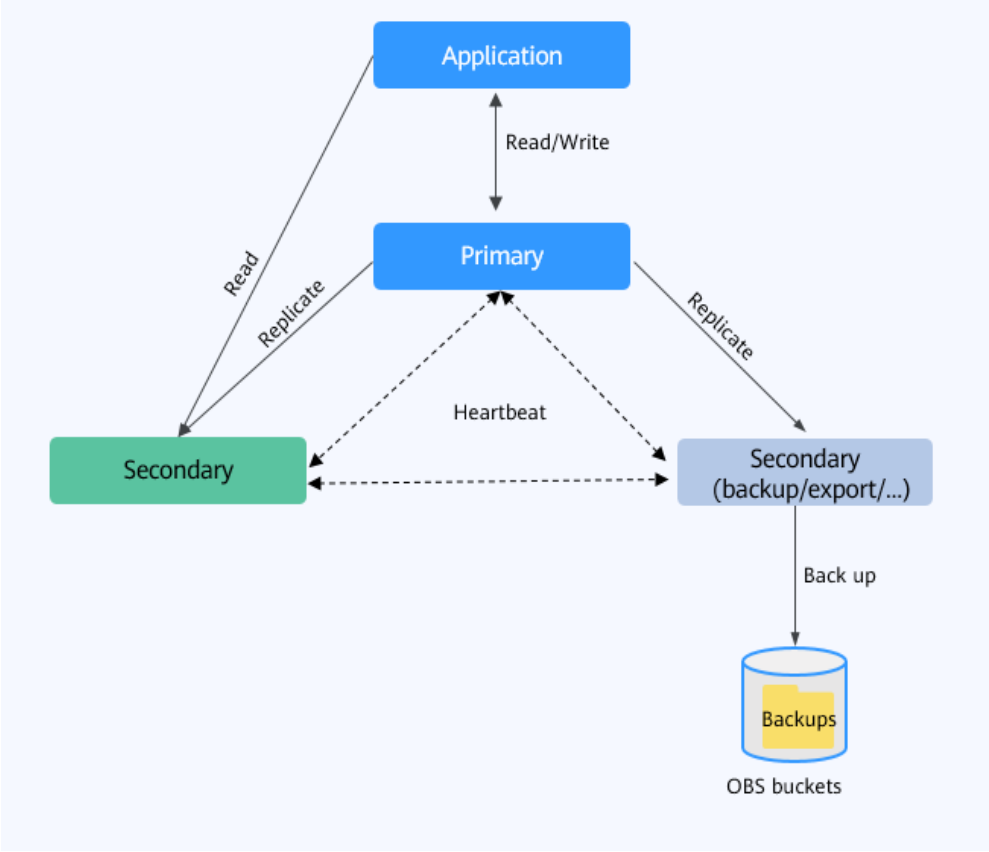
Backup Method	Scenario
Automated backup	After you set a backup policy, the system automatically backs up your instance based on the policy. You can also modify the policy based on service requirements.
Manual backup	You can manually create full backups for your instance based on service requirements.

Backup Principles

GeminiDB Mongo supports only replica set instances.

Each replica set instance can contain one primary node and multiple secondary nodes. As shown in **Figure 4-38**, the GeminiDB Mongo replica set is backed up on the node with the minimum dictionary order ID. Data backup will consume CPU and memory resources of the node, so the CPU and memory usage may increase. Final backups will be compressed and stored in OBS, without occupying storage space of the instance.

Figure 4-38 Backup flowchart



Backup Storage

Backups are stored in OBS buckets to provide disaster recovery and save storage space.

After you purchase an instance, GeminiDB Mongo API will provide additional backup storage of the same size as you purchased. For example, if you purchase an instance of 100 GB, you will obtain additional backup storage of 100 GB free of charge. If the size of backup data does not exceed 100 GB, the backup data is stored on OBS free of charge. If the size of the backup data exceeds 100 GB, you will be charged based on the OBS billing rules.

4.7.2 Managing Automated Backups

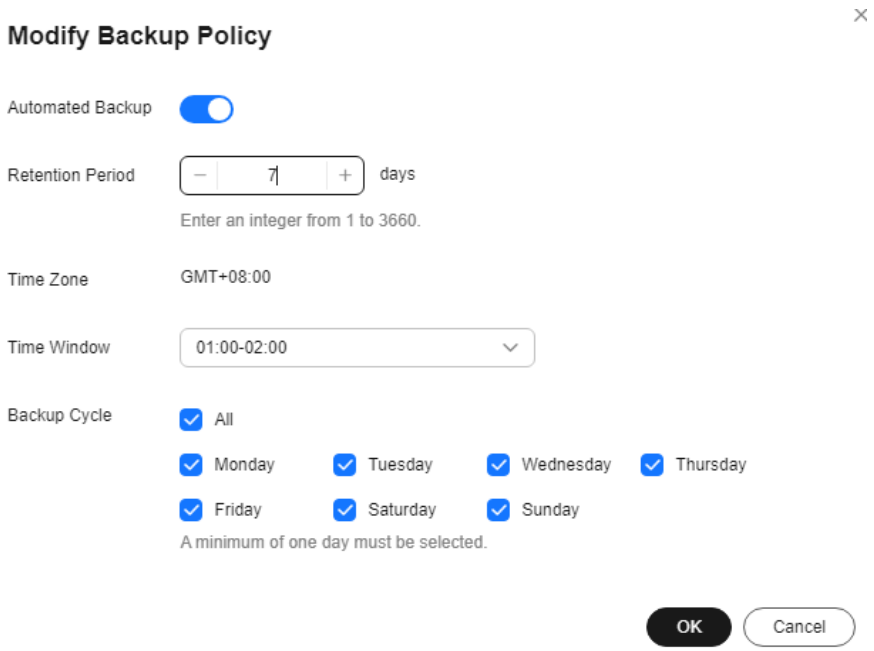
GeminiDB Mongo API supports automated backups to ensure data reliability. If a database or table is deleted, maliciously or accidentally, backups can help recover your data.

Configuring an Automated Backup Policy

Automated backups are generated according to a backup policy and saved as packages in OBS buckets to ensure data confidentiality and durability. You are advised to regularly back up your database, in case it becomes faulty or damaged. Backing up data affects the database read and write performance, so you are advised to set the automated backup time window to off-peak hours.

When you create an instance, automated backup is enabled by default.

Figure 4-39 Modifying a backup policy



- **Retention Period**: Automated backup files are saved for seven days by default. The backup retention period can range from 1 to 35 days. Full backups are retained till the retention period expires.

  - Extending the retention period improves data reliability. You can extend the retention period as needed.
  - If you shorten the retention period, the new backup policy takes effect for existing backups. Any automated backups (including full and incremental backups) that have expired will be automatically deleted. Manual backups will not be automatically deleted but you can delete them manually.

  NOTE

  - If the retention period is less than seven days, the system automatically backs up data daily.
  - The system checks existing automated backup files and deletes the files that exceed the backup retention period you set.
  - **Time Window**: A one-hour period the backup will be scheduled within 24 hours, such as 04:00–05:00. The backup time is in GMT format. If the DST or standard time is switched, the backup time segment changes with the time zone.

  If **Retention Period** is set to **2**, full and incremental backups that have been stored for more than two days will be automatically deleted. That is, the backup generated on Monday will be deleted on Wednesday. Similarly, the backup generated on Tuesday will be deleted on Thursday.

  Policy for automatically deleting full backups:

  To ensure data integrity, even after the retention period expires, the most recent backup will be retained, for example,

  If **Backup Cycle** was set to **Monday** and **Tuesday** and the **Retention Period** was set to **2**:

  - The full backup generated on Monday will be automatically deleted on Thursday. The reasons are as follows:

    The backup generated on Monday expires on Wednesday, but it is the last backup, so it will be retained until a new backup expires. The next backup will be generated on Tuesday and will expire on Thursday. So the full backup generated on Monday will not be automatically deleted until Thursday.
  - The full backup generated on Tuesday will be automatically deleted on the following Wednesday. The reasons are as follows:

    The backup generated on Tuesday will expire on Thursday, but as it is the last backup, so it will be retained until a new backup expires. The next backup will be generated on the following Monday and will expire on the following Wednesday. So the full backup generated on Tuesday will not be automatically deleted until the following Wednesday.

- **Backup Cycle**: All options are selected by default.
  - **All**: Each day of the week is selected. The system automatically backs up data every day.
  - Select a cycle: You can select one or more days in a week. The system automatically backs up data at the specified time.

  NOTE

  A full backup starts within one hour of the time you specify. The amount of time required for the backup depends on the amount of data to be backed up. The more data has to be backed up, the longer it will take.

- After the DB instance is created, you can modify the automated backup policy as needed. You can change the time window after the DB instance is created. The system backs up data based on the automated backup policy you have set.
- If automated backup is disabled, any automated backups in progress stop immediately.

Modifying an Automated Backup Policy

Step 1 Log in to the management console.

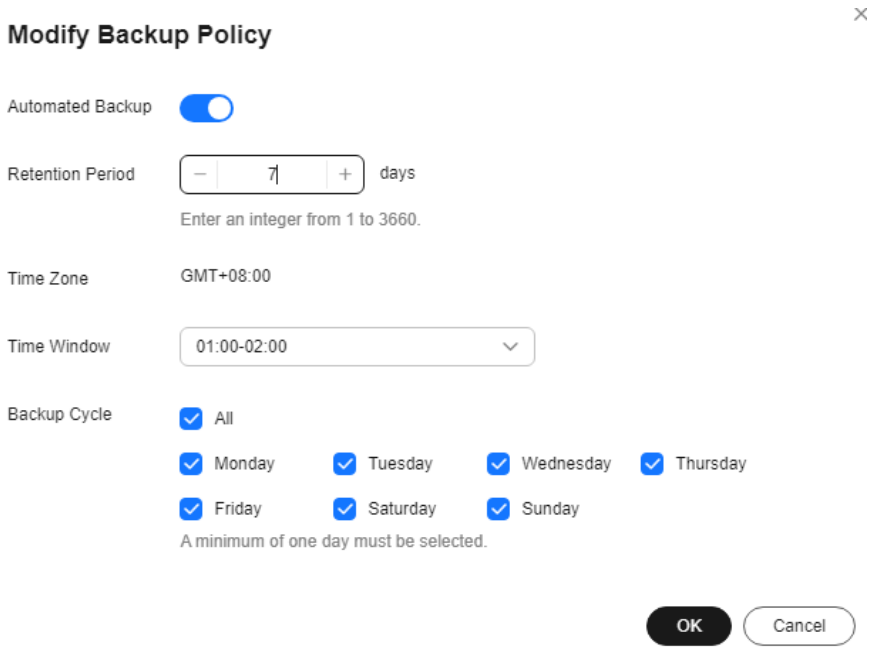
Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance whose backup policy you want to modify.

Step 4 Choose **Backups & Restorations** in the navigation pane one the left, and click **Modify Backup Policy**. In the displayed dialog box, set the backup policy. Then, click **OK**.

For details about how to set a backup policy, see **Configuring an Automated Backup Policy**.

Figure 4-40 Modifying backup policies



Step 5 Check or manage the generated backups on the **Backups** or **Backups & Restorations** page.

----End

Disabling Automated Backup

Step 1 Log in to the management console.

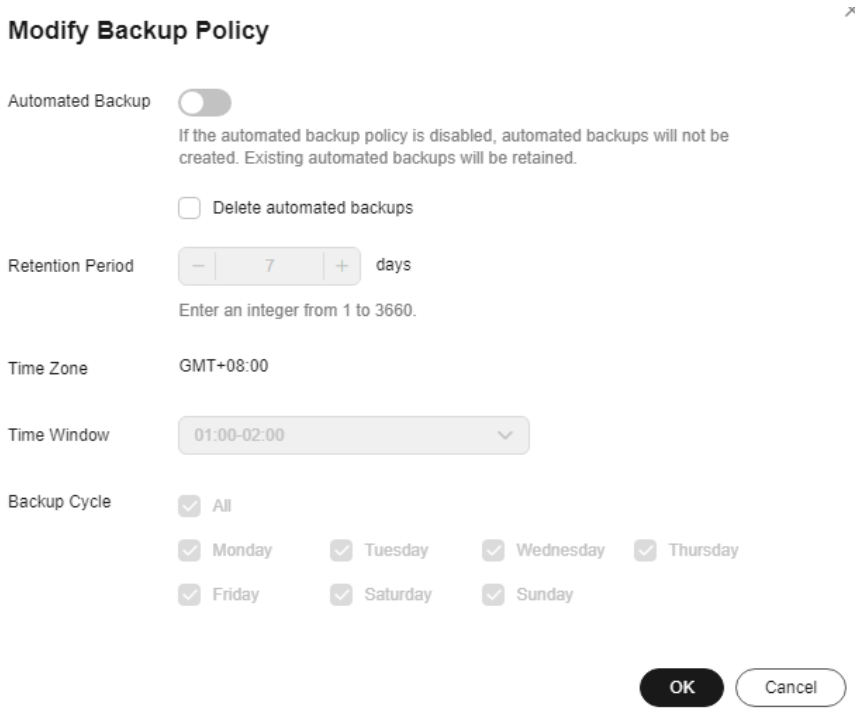
Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click the instance whose backup policy you want to modify.

Step 4 Choose **Backups & Restorations** in the navigation pane one the left, and click **Modify Backup Policy**.

Step 5 In the displayed dialog box, click  and click **OK**.

Figure 4-41 Disabling the automated backup policy



When your disable automated backup, specify whether to delete the automated backups:

- If you select **Delete automated backups**, all backup files within the retention period will be deleted. There are no automated backups displayed until you enable automated backup again.
- If you do not select **Delete automated backups**, backup files within the retention period will be retained, but you can still manually delete them later if needed. For details, see **Deleting an Automated Backup**.

If automated backup is disabled, any automated backups in progress stop immediately.

----End

Deleting an Automated Backup

If automated backup is disabled, you can delete stored automated backups to free up storage space.

If automated backup is enabled, the system will delete automated backups when they expire. You cannot delete them manually.

NOTICE

Deleted backups cannot be recovered. Exercise caution when performing this operation.

- **Method 1**
  a. Log in to the management console.
  b. In the service list, choose **Databases** > **GeminiDB Mongo API**.
  c. On the **Instances** page, click the instance whose backup you want to delete.
  d. Choose **Backups & Restorations** in the navigation pane on the left, locate the backup you want to delete, and click **Delete** in the **Operation** column.
  e. In the displayed dialog box, confirm the backup details and click **Yes**.
- **Method 2**
  a. Log in to the management console.
  b. In the service list, choose **Databases** > **GeminiDB Mongo API**.
  c. On the **Backups** page, locate the backup and click **Delete**.
  d. In the **Delete Backup** dialog box, confirm the backup details and click **Yes**.

4.7.3 Managing Manual Backups

To ensure data reliability, GeminiDB Mongo API allows you to manually back up instances whose status is **Available**. If a database or table is deleted, maliciously or accidentally, backups can help recover your data.

Precautions

- Manual backups are full backups.

Creating a Manual Backup

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 Create a manual backup.

Method 1

On the **Instances** page, locate the instance you want to back up and choose **More** > **Create Backup** in the **Operation** column.

Method 2

1. On the **Instances** page, click the instance you want to create a backup for.

2. Choose **Backups & Restorations** in the navigation pane on the left, and click **Create Backup**.

Method 3

In the navigation pane on the left, choose **Backups**. On the displayed page, click **Create Backup**.

Step 4 In the displayed dialog box, specify a backup name and description and click **OK**.

Figure 4-42 Creating a manual backup

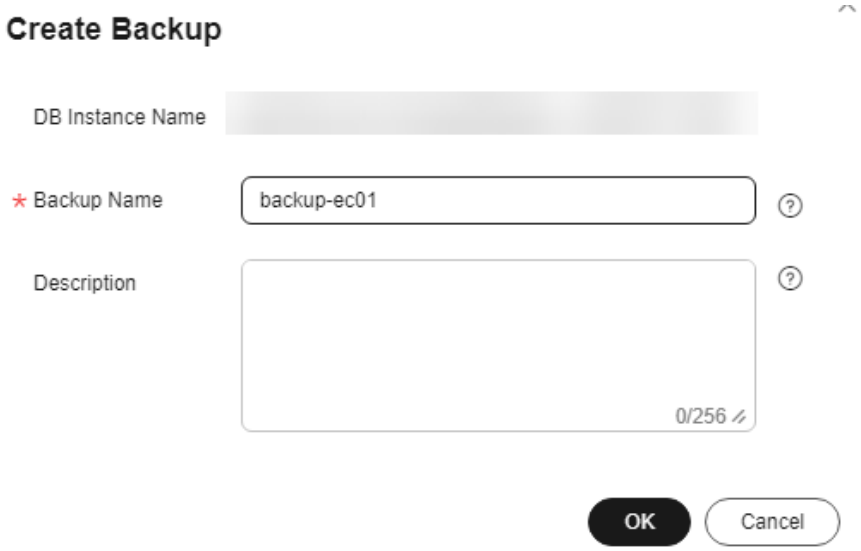


Table 4-14 Parameter description

Parameter	Description
DB Instance Name	Must be the name of the DB instance to be backed up and cannot be modified.
Backup Name	Must be 4 to 64 characters long and start with a letter. It is case-insensitive and contains only letters, digits, hyphens (-), and underscores (_).
Description	Can include a maximum of 256 characters and cannot include line breaks or special characters >!<"&'=

Step 5 View the backup status.

- When the backup is being created, query the backup status on the **Backups** or **Backups & Restorations** page. The backup status is **Backing up**.
- After the backup is created, the backup status changes to **Completed**.

----End

Deleting a Manual Backup

If you do not need a manual backup any longer, delete it on the **Backups** or **Backups & Restorations** page.

Deleted backups are not displayed in the backup list.

NOTICE

Deleted backups cannot be recovered. Exercise caution when performing this operation.

Method 1

1. Log in to the management console.
2. In the service list, choose **Databases** > **GeminiDB Mongo API**.
3. On the **Instances** page, click the instance whose backup you want to delete and click its name.
4. Choose **Backups & Restorations** in the navigation pane on the left, locate the backup you want to delete, and click **Delete** in the **Operation** column.
5. In the displayed dialog box, confirm the backup details and click **Yes**.

Method 2

1. Log in to the management console.
2. In the service list, choose **Databases** > **GeminiDB Mongo API**.
3. On the **Backups** page, locate the backup that you want to delete and click **Delete**.
4. In the displayed dialog box, confirm the backup details and click **Yes**.

4.8 Data Restoration

4.8.1 Restoration Methods

GeminiDB Mongo API supports multiple forms of data restoration. You can select one based on service requirements.

Table 4-15 Restoration methods

Method	Scenario
Restoring data to a new instance	You can restore an existing backup file to a new instance.
Restoring data to an existing instance	You can restore an existing backup file to an existing instance.

4.8.2 Restoring Data to a New or Existing Instance

Scenarios

GeminiDB Mongo API allows you to use an existing backup to restore data to a new or existing instance.

Precautions

This function is in the open beta test (OBT) phase. To use this function, contact customer service.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 Restore a DB instance from the backup.

Method 1

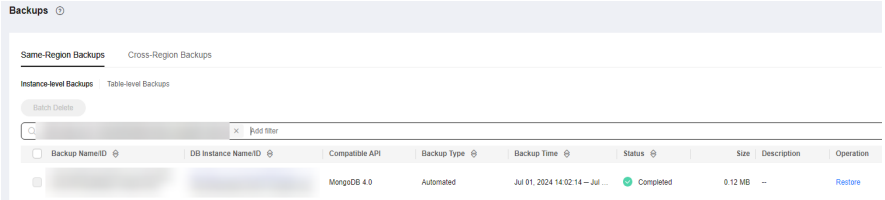
1. On the **Instances** page, locate the instance whose backup you want to restore and click its name.
2. On the **Backups & Restorations** page, locate the target backup and click **Restore**.

Figure 4-43 Backups and restorations

Method 2

On the **Backups** page, locate the backup that you want to restore and click **Restore** in the **Operation** column.

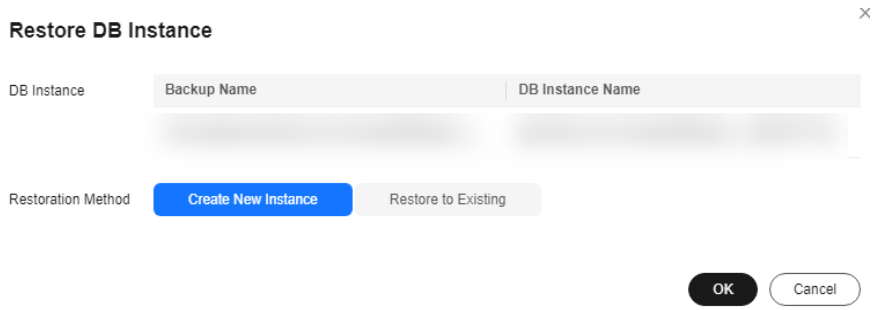
Figure 4-44 Backup management



Step 4 In the displayed dialog box, confirm the instance details and restoration method and click **OK**.

- Create New Instance

Figure 4-45 Restoring data to a new instance



The **Create New Instance** page is displayed. Configure the new DB instance details based on your service requirements.

- The default API type and DB engine version are the same as those of the original instance and cannot be changed.
- GeminiDB automatically calculates the minimum storage space required for restoration based on the size of the selected backup file. The storage capacity depends on the instance specifications, and must be an integer.
- You need to set a new administrator password.
- To modify other parameters, see the description of buying DB instances of other DB engines in the *Getting Started with GeminiDB Mongo*.

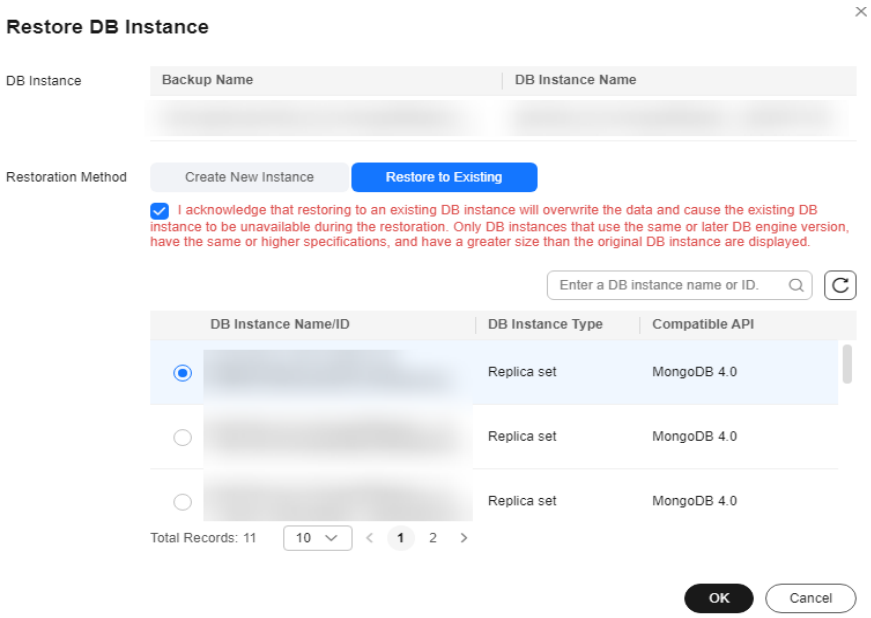
- Restore to Existing

  Select an existing instance and click **OK**.

NOTICE

- You can restore a deleted instance to an existing instance from a backup if available.
- Restoring to an existing DB instance will overwrite the data, and the existing DB instance becomes unavailable during the restoration.
- After data is restored to an existing DB instance, the username and password saved in the backup file will overwrite those of the existing DB instance.
- Only DB instances that have the same number of nodes, use the same or later DB API version, and have the same or higher specifications than the original DB instance can be selected.
- The storage space of the instance that you want to restore the backup to must be greater than or equal to the size of the backup. Otherwise, the restoration task fails.

Figure 4-46 Restore DB Instance



Step 5 View the restoration results.

- Create New Instance

  A new instance is created using the backup data. The restoration is successful if the status of the new instance changes from **Creating** to **Available**.

  After the restoration is complete, a full backup will be automatically triggered.

  The new DB instance is independent from the original one.

- Restore to Existing

  On the **Instances** page, the status of the instance changes from **Restoring** to **Available**.

  If automated backup is enabled, a full backup will be triggered after the restoration is complete. Otherwise, the full backup will not be triggered.

----End

4.9 Parameter Template Management

4.9.1 Creating a Parameter Template

You can use database parameter templates to manage DB API configurations. A database parameter template acts as a container for API configuration values that can be applied to one or more DB instances.

Each user can create up to 100 parameter templates. All types of instances in the same project can share the quota.

Precautions

This function is in the open beta test (OBT) phase. Contact customer service to apply for the function.

Procedure

Step 1 Log in to the management console.

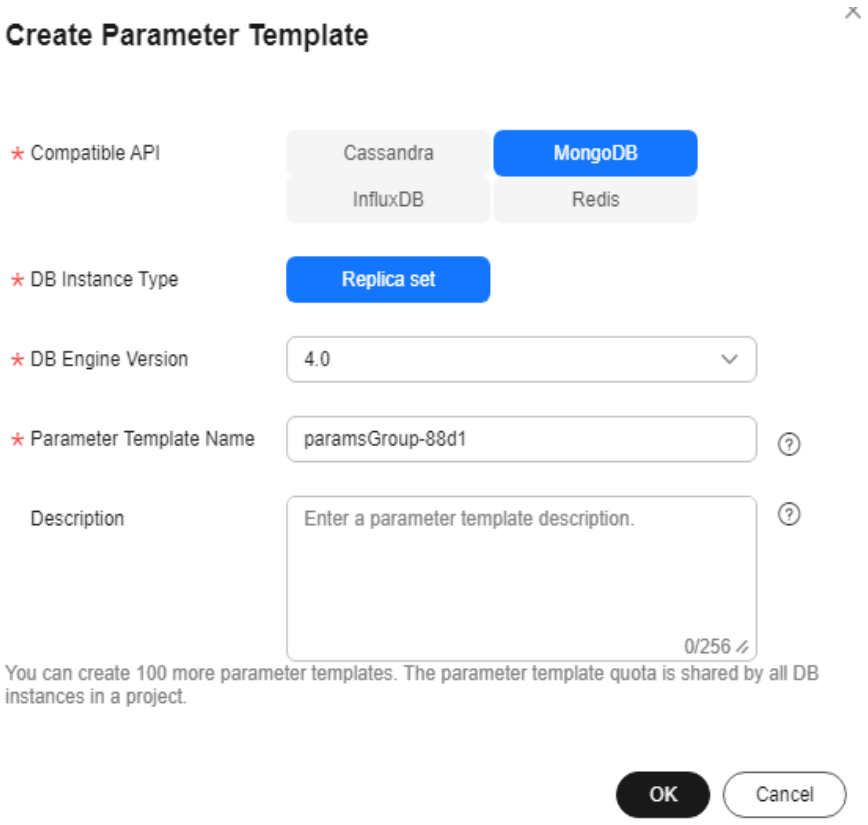
Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, click **Create Parameter Template**.

Step 5 Select a compatible API, specify a DB engine version and a parameter group description, and click **OK**.

Figure 4-47 Creating a parameter template



- **Compatible API**: Select the API type that is compatible with your DB API parameter template.
- **DB Instance Type**: Select an instance type.
- **DB Engine Version**: Select **4.0**.
- **Parameter Template Name**: The template name can be up to 64 characters long. It can contain only uppercase letters, lowercase letters, digits, hyphens (-), underscores (_), and periods (.).

- **Description**: The description contains a maximum of 256 characters and cannot include line breaks or the following special characters >!<"&'=

Step 6 On the **Parameter Templates** page, view the created parameter template.

----End

4.9.2 Modifying Parameters of GeminiDB Mongo Instances

You can modify parameters in a custom parameter template so that your instance can deliver spectacular performance.

Note that parameter values in default parameter templates cannot be changed.

NOTE

Though parameter values in a default template cannot be changed, you can view details about a default parameter template. If a custom parameter template is set incorrectly, the database startup may fail. You can re-configure the custom parameter template according to the configurations of the default parameter template.

Modifying a Custom Parameter Template and Applying It to an Instance

Step 1 Log in to the management console.

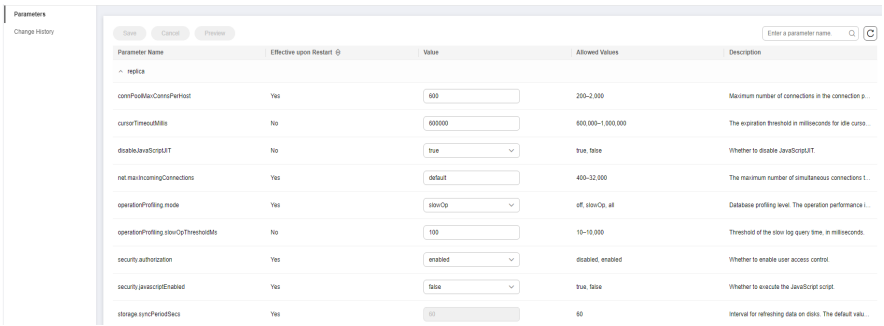
Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Custom Templates** page, click the parameter template you want to modify.

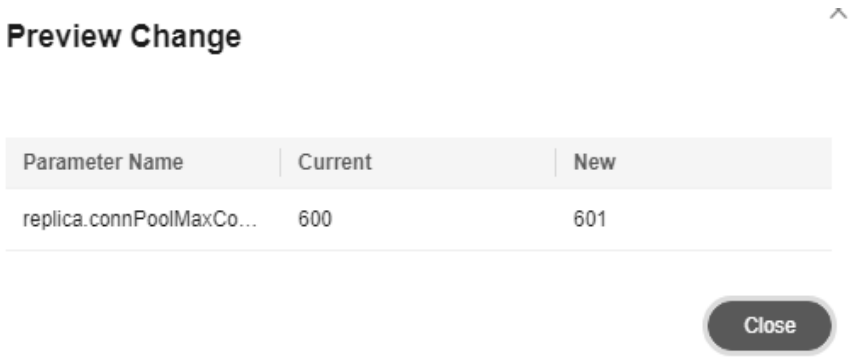
Step 5 Change parameter values as required.

Figure 4-48 Modifying parameters in a parameter template



- To save the modifications, click **Save**.
- To cancel the modifications, click **Cancel**.
- To preview the modifications, click **Preview**.

Figure 4-49 Previewing changes



Step 6 After parameters are modified, click **Change History** to view parameter modification details.

For details about how to view parameter modification details, see **Viewing Parameter Change History**.

NOTICE

- The modifications take effect only after you apply the parameter template to DB instances. For details, see **Applying a Parameter Template**.
- The change history page displays only the modifications of the last seven days.

----End

Modifying Parameters of an Instance

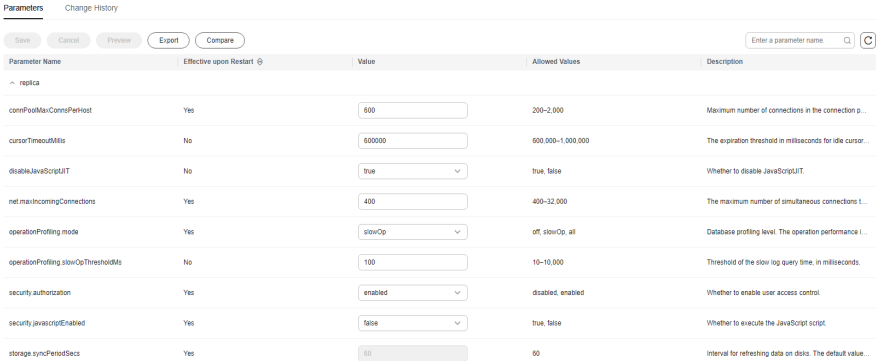
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Instances**, locate the instance whose parameters you want to export, and click its name.

Step 4 In the navigation pane on the left, choose **Parameters**. On the displayed page, modify parameters as required.

Figure 4-50 Parameters



- To save the modifications, click **Save**.
- To cancel the modifications, click **Cancel**.
- To preview the modifications, click **Preview**.

Step 5 After parameters are modified, click **Change History** to view parameter modification details.

For details about how to view parameter modification details, see **Viewing Parameter Change History**.

NOTICE

After you modify instance parameters, the modifications immediately take effect for the instance.

Check the value in the **Effective upon Restart** column.

- If the value is **Yes** and the instance status is **Pending restart**, restart the instance for the modifications to take effect.
- If the value is **No**, the modifications take effect immediately.

----End

4.9.3 Viewing Parameter Change History

Scenarios

You can view parameter change history of an instance or one of its custom parameter templates based on service requirements.

NOTE

In a newly exported or created parameter template, change history is left blank.

Viewing Change History of a Custom Parameter Template

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**. On the **Custom Templates** page, click the parameter template you want to view.

Step 4 In the navigation pane on the left, choose **Change History**. Then, view the name, original value, new value, modification status, and modification time of the target parameter.

Figure 4-51 Viewing change history of a customer parameter template

You can apply the parameter template to instances by referring to **Applying a Parameter Template**.

----End

Viewing Parameter Change History of an Instance

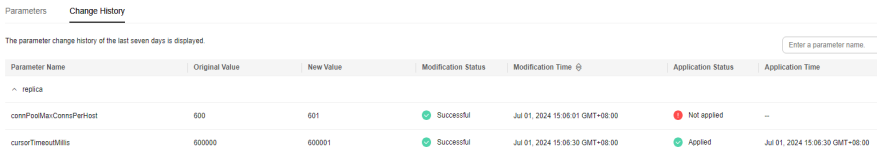
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose parameter change history you want to view and click its name.

Step 4 In the navigation pane on the left, choose **Parameters**. On the **Change History** page, view the name, original value, new value, modification status, and modification time of the target parameter.

Figure 4-52 Viewing parameter change history of an instance



----End

4.9.4 Exporting a Parameter Template

Scenarios

- You can export a parameter template of a DB instance for future use. To learn how to apply the exported parameter template to a DB instance, refer to section **Applying a Parameter Template**.
- You can export the parameter template details (parameter names, values, and descriptions) of a DB instance to a CSV file for review and analysis.

Procedure

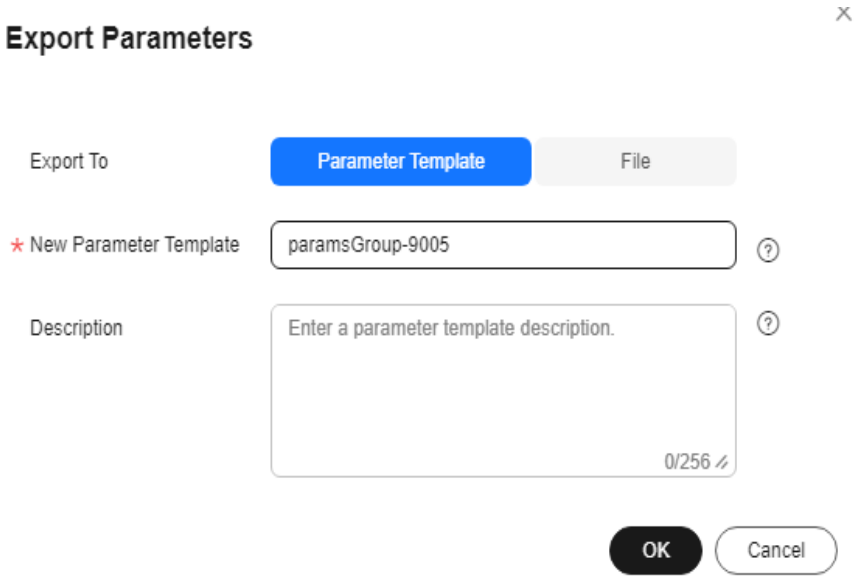
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Instance Management**. On the displayed page, click the target DB instance.

Step 4 In the navigation pane on the left, choose **Parameters**. On the **Parameters** tab, above the parameter list, click **Export**.

Figure 4-53 Exporting a parameter template



- **Parameter Template**: You can export the parameters of the DB instance to a template for future use.

  In the displayed dialog box, configure required details and click **OK**.

  NOTE

  - The template name can include 1 to 64 characters. It can contain only uppercase letters, lowercase letters, digits, hyphens (-), underscores (_), and periods (.).
  - The template description consists of a maximum of 256 characters and cannot include line breaks or the following special characters: >!<"&'=

  After the parameter template is exported, a new template is generated in the list on the **Parameter Templates** page.

- **File**: You can export the parameter template details (parameter names, values, and descriptions) of a DB instance to a CSV file for review and analysis.

  In the displayed dialog box, enter the file name and click **OK**.

  NOTE

  The file name must start with a letter and can include 4 to 81 characters. It can contain only letters, digits, hyphens (-), and underscores (_).

----End

4.9.5 Comparing Parameter Templates

Scenarios

This section describes how to compare two parameter templates of the same instance type and compatible API to learn about their configurations.

Comparing Parameter Templates

Step 1 Log in to the management console.

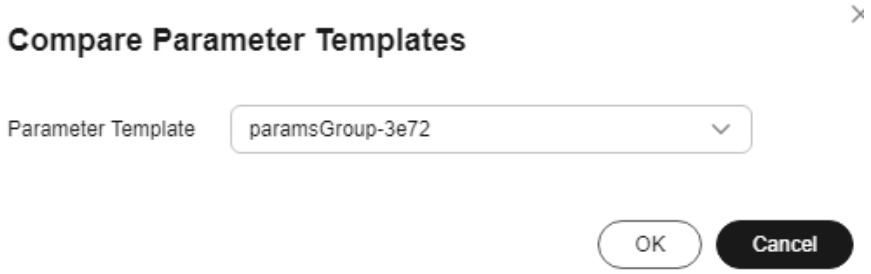
Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 In the parameter template list, locate the parameter template that you created and click **Compare** in the **Operation** column.

Step 5 In the displayed dialog box, select a parameter template that is of the same instance type and compatible API as the selected template and click **OK**.

Figure 4-54 Comparing two parameter templates



- If their parameters are different, the different parameter names and values are displayed.
- If their parameters are the same, no data is displayed.

----End

Comparing Parameter Templates of an Instance

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

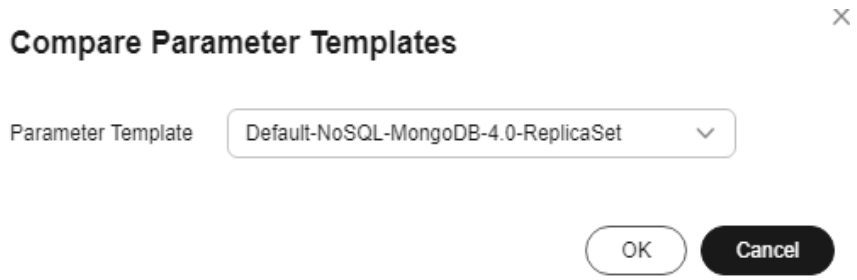
Step 3 In the navigation pane on the left, choose **Instances**.

Step 4 In the instance list, locate the instance whose parameter templates you want to compare and click its name.

Step 5 In the navigation pane on the left, choose **Parameters** and then click **Compare** above the parameter list.

Step 6 In the displayed dialog box, select a parameter template that is of the same instance type as the template of current instance and click **OK**.

Figure 4-55 Comparing the instance parameter template with another parameter template



- If their parameters are different, the different parameter names and values are displayed.
- If their parameters are the same, no data is displayed.

----End

4.9.6 Replicating a Parameter Template

Scenarios

You can replicate a parameter template you have created. When you have already created a parameter template and want to include most of the custom parameters and values from that template in a new parameter template, you can replicate that parameter template. You can also export a parameter template of a DB instance for future use.

Default parameter templates cannot be replicated. You can create parameter templates based on the default ones.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

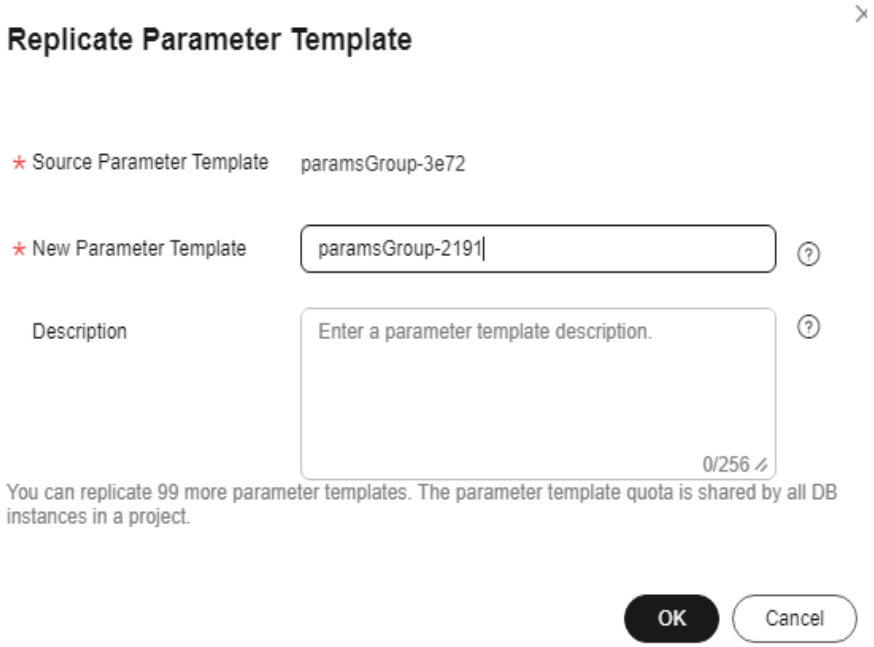
Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, click the **Custom Templates** tab. Locate the target parameter template and click **Replicate** in the **Operation** column.

Alternatively, click the target DB instance on the **Instances** page. On the **Parameters** page, click **Export** to generate a new parameter template for future use.

Step 5 In the displayed dialog box, enter the parameter template name and description and click **OK**.

Figure 4-56 Replicate Parameter Template



- **New Parameter Template**: The template name can be up to 64 characters long. It can contain only uppercase letters, lowercase letters, digits, hyphens (-), underscores (_), and periods (.).
- **Description**: The description contains a maximum of 256 characters and cannot include line breaks or the following special characters >!<"&'=

After the parameter template is replicated, a new template is generated in the list on the **Parameter Template Management** page.

----End

4.9.7 Resetting a Parameter Template

Scenarios

You can reset all parameters in a custom parameter template to their default settings.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, click the **Custom Templates** tab. Locate the target parameter template and choose **More** > **Reset** in the **Operation** column.

Step 5 Click **Yes** to reset the parameter template.

----End

4.9.8 Applying a Parameter Template

Scenarios

GeminiDB Mongo allows you to apply a parameter template. Modifications to parameters in a custom parameter template take effect only after you have applied the template to the target instance.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, perform the following operations based on the template type:

- To apply a default template, click **Default Templates**, locate the template, and in the **Operation** column, click **Apply**.
- To apply a custom template, click **Custom Templates**, locate the template, and in the **Operation** column, choose **More** > **Apply**.

A parameter template can be applied to one or more instances.

Step 5 In the displayed dialog box, select one or more instances that the parameter template will be applied to and click **OK**.

After a parameter template is applied, you can **view its application records**.

----End

4.9.9 Viewing Application Records of a Parameter Template

Scenarios

GeminiDB Mongo allows you to view application records of a parameter template.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, locate the parameter template and choose **More** > **View Application Record** in the **Operation** column.

You can view the name or ID of the instance that the parameter template applies to, as well as the application status, application time, and causes of any failures that have occurred.

----End

4.9.10 Modifying a Parameter Template Description

Scenarios

You can modify the description of a custom parameter template if needed.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, click the **Custom Templates** tab. Locate the target parameter template and click  in the **Description** column.

Step 5 Enter a new description. You can click  to submit or  to cancel the modification.

- After you submit the modification, you can view the new description in the **Description** column on the **Parameter Templates** page.
- The description can include up to 256 characters but cannot contain the following special characters: >!<"&'=

----End

4.9.11 Deleting a Parameter Template

Scenarios

You can delete a custom parameter template that is no longer in use.

Precautions

- Deleted templates cannot be recovered, so exercise caution when performing this operation.
- Default parameter templates cannot be deleted.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane on the left, choose **Parameter Templates**.

Step 4 On the **Parameter Templates** page, click **Custom Templates**. Locate the parameter template you want to delete and choose **More** > **Delete** in the **Operation** column.

Step 5 Click **Yes** to delete the parameter template.

----End

4.10 Monitoring and Alarm Configuration

4.10.1 GeminiDB Mongo Metrics

Description

This section describes GeminiDB Mongo metrics reported to Cloud Eye as well as their namespaces and dimensions. You can use APIs provided by Cloud Eye to query the metrics of the monitored object and alarms generated.

Namespace

SYS.NoSQL

Metrics

NOTE

You can view metrics on instance nodes by referring to **Viewing Metrics**.

Table 4-16 GeminiDB Mongo metrics

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
nosql001_cpu_usage	CPU Usage	CPU usage of the monitored system Unit: Percent	0–100	GeminiDB Mongo instance nodes	1 minute
nosql002_mem_usage	Memory Usage	Memory usage of the monitored system Unit: Percent	0–100	GeminiDB Mongo instance nodes	1 minute
nosql003_bytes_out	Network Output Throughput	Outgoing traffic in bytes per second Unit: bytes/s	≥ 0	GeminiDB Mongo instance nodes	1 minute

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
nosql004_bytes_in	Network Input Throughput	Incoming traffic in bytes per second Unit: bytes/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
nosql005_disk_usage	Storage Space Usage	Storage space usage of the monitored object. Unit: Percent	0–100	GeminiDB Mongo instances	1 minute
nosql006_disk_total_size	Total Storage Space	Total storage space of the monitored object. Unit: GB	≥ 0	GeminiDB Mongo instances	1 minute
nosql007_disk_used_size	Used Storage Space	Used storage space of the monitored object. Unit: GB	≥ 0	GeminiDB Mongo instances	1 minute
mongodb001_command_ps	COMMAND Statements per Second	Number of COMMAND statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
mongodb002_delete_ps	DELETE Statements per Second	Number of DELETE statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb003_insert_ps	INSERT Statements per Second	Number of INSERT statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb004_query_ps	QUERY Statements per Second	Number of QUERY statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb005_update_ps	UPDATE Statements per Second	Number of UPDATE statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb006_getmore_ps	GETMORE Statements per Second	Number of GETMORE statements executed per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
mongodb007_connections	Current Active Connections	Total number of connections attempting to connect to DB instance nodes Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb007_connections_usage	Percentage of Active Node Connections	Percentage of the number of connections that attempt to connect to the instance node to the total number of available connections Unit: Percent	0–100	GeminiDB Mongo instance nodes	1 minute
mongodb008_mem_resident	Resident Memory	Size of resident memory The unit is MB.	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb009_mem_virtual	Virtual Memory	Size of the virtual memory The unit is MB.	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb010_regular_asserts_ps	Regular Asserts per Second	Number of regular asserts per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
mongodb011_warning_asserts_ps	Warning Asserts per Second	Number of warning asserts per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb012_msg_asserts_ps	Message Asserts per Second	Number of message asserts per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb013_user_asserts_ps	User Asserts per Second	Number of user asserts per second Unit: count/s	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb014_queues_total	Operations Queued Waiting for a Lock	Number of operations queued waiting for the lock Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb015_queues_readers	Operations Queued Waiting for a Read Lock	Number of operations queued waiting for the read lock Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb016_queues_writers	Operations Queued Waiting for a Write Lock	Number of operations queued waiting for the write lock Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute

Metric ID	Metric Name	Description	Value Range	Monitored Object	Monitoring Period (Raw Data)
mongodb017_page_faults	Page Faults	Number of page faults on the monitored nodes Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb018_porfling_num	Slow Queries	Number of slow queries on the monitored nodes Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb019_cursors_open	Maintained Cursors	Number of maintained cursors on the monitored nodes Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute
mongodb020_cursors_timeout	Timeout Cursors	Number of timed out cursors on the monitored nodes Unit: count	≥ 0	GeminiDB Mongo instance nodes	1 minute

Dimensions

Key	Value
mongodb_cluster_id	ID of the GeminiDB Mongo instance
mongodb_node_id	Node ID of the GeminiDB Mongo instance

4.10.2 Configuring Alarm Rules

Scenarios

Setting alarm rules allows you to customize objects to be monitored and notification policies so that you can closely monitor your instances.

Alarm rules include the alarm rule name, instance, metric, threshold, monitoring interval, and whether to send notifications. This section describes how to set alarm rules.

Procedure

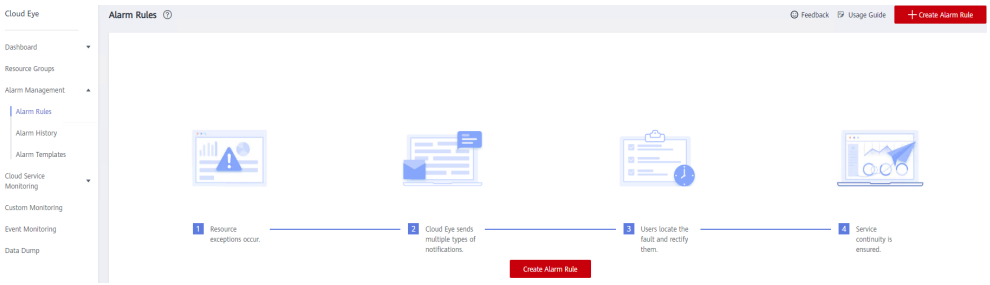
Step 1 Log in to the management console.

Step 2 Click **Service List**. Under **Management & Governance**, click **Cloud Eye**.

Step 3 In the navigation pane on the left, choose **Alarm Management** > **Alarm Rules**.

Step 4 On the **Alarm Rules** page, click **Create Alarm Rule**.

Figure 4-57 Creating an alarm rule



Step 5 Set alarm parameters.

1. Configure basic alarm information.

Figure 4-58 Configuring basic information for an alarm rule

Table 4-17 Basic alarm rule information

Parameter	Description	Example Value
Name	Name of the rule. The system generates a random name and you can modify it.	alarm-cag2
Description	(Optional) Alarm rule description.	-

2. Select objects to be monitored and specify the monitoring scope.

Table 4-18 Parameter description

Parameter	Description	Example Value
Alarm Type	Alarm type that the alarm rule is created for. The value can be **Metric** or **Event**.	Metric
Resource Type	Type of the resource the alarm rule is created for. Select **GeminiDB**.	-
Dimension	Metric dimension of the alarm rule. Select **MongoDB-MongoDB node**.	-
Monitoring Scope	Monitoring scope the alarm rule applies to. **NOTE** – If you select **All resources**, an alarm notification will be sent when any instance meets an alarm policy, and existing alarm rules will be automatically applied for newly purchased resources. – If you select **Resource groups** and any resource in the group meets the alarm policy, an alarm notification will be sent. – To specify **Specific resources**, click **Select Specified Resources**, select one or more resources, and click **OK**.	All Resources
Group	This parameter is mandatory when **Monitoring Scope** is set to **Resource groups**.	-

3. Configure an alarm policy.

Figure 4-59 Configuring the alarm policy

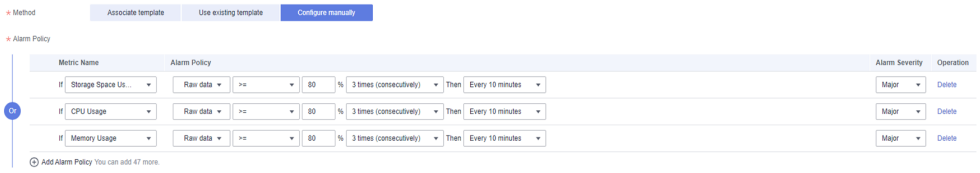


Table 4-19 Parameter description

Parameter	Description	Example Value
Method	Select **Associate template**, **Use existing template**, or **Configure manually**. **NOTE** If you set **Monitoring Scope** to **Specific resources**, you can set **Method** to **Use existing template**.	Configure manually
Template	Select the template to be used. This parameter is available only when you set **Method** to **Use existing template**.	-

Parameter	Description	Example Value
Alarm Policy	Policy for triggering an alarm. You can configure the threshold, consecutive periods, alarm interval, and alarm severity based on service requirements. – **Metric Name**: specifies the the metric that the alarm rule is created for. The following metrics are recommended: **Storage Space Usage**, which is used to monitor the storage usage of GeminiDB Cassandra instances. If the storage usage is greater than 80%, scale up the storage in a timely manner by referring to **Scaling Up Storage Space**. **CPU Usage** and **Memory Usage**, which are used to monitor the compute resource usage of each GeminiDB Cassandra instance node. If the CPU usage or memory usage is greater than 80%, you can **add nodes** or **upgrade node specifications** in a timely manner. For more metrics, see **GeminiDB Mongo Metrics**. – **Alarm Severity**: specifies the severity of the alarm. Valid values are **Critical**, **Major**, **Minor**, and **Informational**. **NOTE** A maximum of 50 alarm policies can be added to an alarm rule. If any one of these alarm policies is met, an alarm is triggered.	Take the CPU usage as an example. The alarm policy configured in **Figure 4-59** indicates that a major alarm will be sent to users every 10 minutes if the original CPU usage reaches 80% or above for three consecutive periods.

4. Configure alarm notification information.

Figure 4-60 Configuring alarm notification information

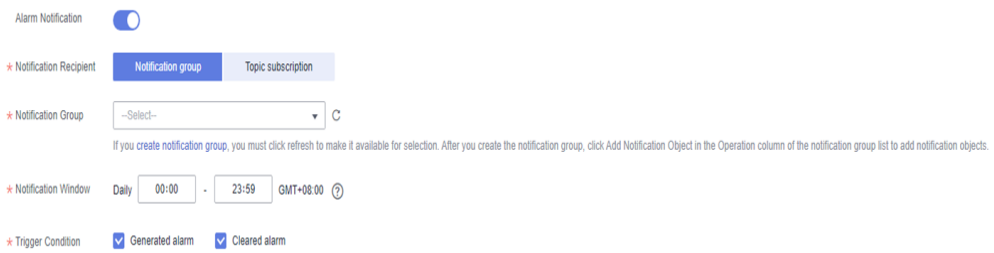


Table 4-20 Parameter description

Parameter	Description	Example Value
Alarm Notification	Whether to notify users when alarms are triggered. Notifications can be sent by email, text message, or HTTP/HTTPS message. Enabling alarm notification is recommended. When the metric data reaches the threshold set in the alarm rule, Cloud Eye immediately notifies you through SMN that an exception has occurred.	Enabled **Alarm Notification**.
Notification Recipient	Select **Notification group** or **Topic subscription**.	-
Notification Group	Notification group the alarm notification is to be sent to.	-
Notification Object	Specifies the object that receives alarm notifications. You can select the account contact or a topic. – Account contact is the mobile phone number and email address provided for registration. – **Topic** is used to publish messages and subscribe to notifications. If the required topic is unavailable, create one first and add subscriptions to it. For details, see **Creating a Topic** and **Adding Subscriptions**.	-
Notification Window	Cloud Eye sends notifications only within the notification window specified in the alarm rule. For example, if **Notification Window** is set to **00:00-8:00**, Cloud Eye sends notifications only within 00:00-08:00.	-

Parameter	Description	Example Value
Trigger Condition	Condition for triggering an alarm notification. You can select **Generated alarm** (when an alarm is generated), **Cleared alarm** (when an alarm is cleared), or both.	-

5. Configure advanced settings.

Figure 4-61 Advanced settings

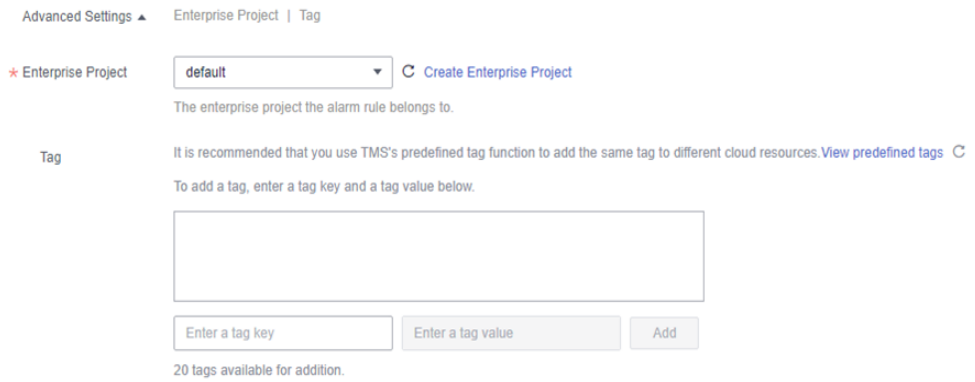


Table 4-21 Parameter description

Parameter	Description	Example Value
Enterprise Project	Enterprise project that the alarm rule belongs to. Only users with the enterprise project permissions can view and manage the alarm rule. For details about how to create an enterprise project, see **Creating an Enterprise Project**.	default
Tag	A tag is a key-value pair. Tags identify cloud resources so that you can easily categorize and search for your resources. You are advised to create predefined tags on TMS. For details about how to create predefined tags, see **Creating Predefined Tags**. – A key can contain a maximum of 128 characters, and a value can contain a maximum of 225 characters. – A maximum of 20 tags can be added.	-

Step 6 After the configuration is complete, click **Create**.

When the metric data reaches the threshold set in the alarm rule, Cloud Eye immediately notifies you through SMN that an exception has occurred.

----End

4.10.3 Viewing Metrics

Scenarios

Cloud Eye monitors statuses of GeminiDB Mongo instances. You can view GeminiDB Mongo metrics on the management console.

Monitored data requires a period of time for transmission and display. The status of the monitored object displayed on the Cloud Eye page is the status obtained 5 to 10 minutes before. You can view the monitored data of a newly created DB instance 5 to 10 minutes later.

Precautions

- The DB instance is running properly.

  Cloud Eye does not display the metrics of a faulty or deleted DB instance. You can view the monitoring information only after the instance is restarted or recovered.
- The DB instance has been properly running for at least 10 minutes.

  The monitoring data and graphics are available for a new instance after the instance runs for at least 10 minutes.

Method 1

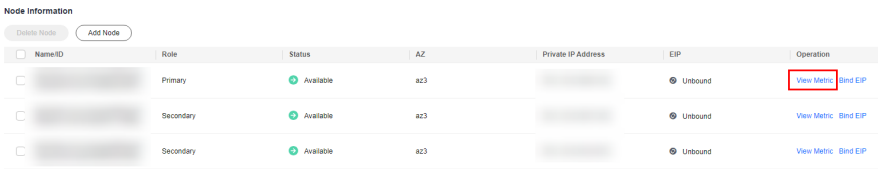
Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instance Management** page, locate the target instance and click its name.

Step 4 In the **Node Information** area on the **Basic Information** page, click **View Metric** in the **Operation** column.

Figure 4-62 Querying monitoring metrics



Step 5 In the monitoring area, you can select a duration to view the monitoring data.

You can view the monitoring data of the service in the last 1, 3, or 12 hours.

To view the monitoring curve in a longer time range, click to enlarge the graph.

----End

Method 2

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

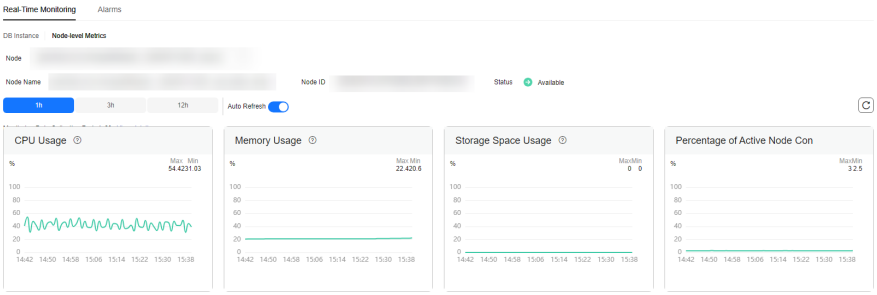
Step 3 On the **Instance Management** page, locate the target instance and click its name.

Step 4 In the navigation pane on the left, choose **Advanced O&M**.

Step 5 On the **Advanced O&M** page, click the **Real-Time Monitoring** tab.

Step 6 Select a node from the **Node** drop-down list, select the duration, and enable **Auto Refresh** to view the real-time monitoring data.

Figure 4-63 Viewing metrics



- You can view the monitoring data of the service in the last 1, 3, or 12 hours.
- The monitoring data on the current page is collected every 60 seconds.
- The current page displays only the monitoring data of common metrics, such as the CPU usage, memory usage, and storage usage. If you want to view more metrics, click **View details** to go to the Cloud Eye console.

----End

4.10.4 Event Monitoring

4.10.4.1 Introduction to Event Monitoring

Event monitoring provides event data reporting, query, and alarm reporting. You can create alarm rules for both system and custom events. When a specific event occurs, Cloud Eye generates and sends an alarm for you.

Key operations on GeminiDB Mongo resources are monitored and recorded by Cloud Eye as events. Events include operations performed by specific users on specific resources, such as changing instance names and specifications.

Event monitoring provides an API for reporting custom events, which helps you collect and report abnormal events or important change events generated by services to Cloud Eye.

Event monitoring is enabled by default and allows you to view monitoring details of system events and custom events. For details about system events, see **Events Supported by Event Monitoring**.

4.10.4.2 Viewing Event Monitoring Data

Scenarios

Event monitoring provides event data reporting, query, and alarm reporting. You can create alarm rules for both system and custom events. When a specific event occurs, Cloud Eye generates and sends an alarm for you.

Event monitoring is enabled by default. You can view monitoring details about system events and custom events.

This topic describes how to view the event monitoring data.

Procedure

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 You can view event monitoring data in either of the following ways:

- In the navigation tree on the left, choose **Advanced O&M**, and then click the **Real-Time Monitoring** tab. Select a node from the **Node** drop-down list, select the duration, and enable **Auto Refresh** to view the real-time monitoring data.
- In the **Node Information** area on the **Basic Information** page, click **View Metric** in the **Operation** column.

Step 4 Click < to return to the Cloud Eye console.

Step 5 In the navigation pane on the left, choose **Event Monitoring**.

On the displayed **Event Monitoring** page, all system events generated in the last 24 hours are displayed by default.

You can also click **1h**, **3h**, **12h**, **1d**, **7d**, or **30d** to view events generated in different time periods.

Step 6 Locate an event and click **View Event** in the **Operation** column to view its details.

----End

4.10.4.3 Creating an Alarm Rule for Event Monitoring

Scenarios

This topic describes how to create an alarm rule for event monitoring.

Procedure

Step 1 **Log in to the management console.**

Step 2 Click ☰ in the upper left corner of the page. Under **Management & Governance**, click **Cloud Eye**.

Step 3 In the navigation pane on the left, choose **Event Monitoring**.

Step 4 On the event list page, click **Create Alarm Rule** in the upper right corner.

Step 5 On the **Create Alarm Rule** page, configure the parameters.

Table 4-22 Parameter description

Parameter	Description
Name	Specifies the name of the alarm rule. The system generates a random name, but you can change it if needed.
Description	(Optional) Provides supplementary information about the alarm rule.
Enterprise Project	You can select an existing enterprise project or click **Create Enterprise Project** to create one.
Alarm Type	Specifies the alarm type corresponding to the alarm rule.
Event Type	Specifies the event type of the metric corresponding to the alarm rule.
Event Source	Specifies the service the event is generated for. Select GeminiDB.
Monitoring Scope	Specifies the monitoring scope for event monitoring.
Method	Specifies the event creation method.
Alarm Policy	**Event Name** indicates the instantaneous operations users performed on system resources, such as login and logout. For details about events supported by Event Monitoring, see **Events Supported by Event Monitoring**. You can select a trigger mode and alarm severity as needed.

Click  to enable alarm notification. The validity period is 24 hours by default. If the topics you require are not displayed in the drop-down list, click **Create an SMN topic**.

Table 4-23 Alarm notification parameters

Parameter	Description
Alarm Notification	Specifies whether to notify users when alarms are triggered. Notifications can be sent by email, text message, or HTTP/HTTPS message.

Parameter	Description
Notification Object	Specifies the object an alarm notification is to be sent to. You can select the account contact or a topic. • Account contact is the mobile phone number and email address provided for registration. • Topic is used to publish messages and subscribe to notifications. If the required topic is unavailable, create one first and add subscriptions to it. For details, see **Creating a Topic** and **Adding Subscriptions**.
Validity Period	Cloud Eye sends notifications only within the validity period specified in the alarm rule. If you set **Validity Period** to **08:00-20:00**, Cloud Eye sends notifications only within 08:00-20:00.
Trigger Condition	Specifies the condition for triggering the alarm notification.

Step 6 After the configuration is complete, click **Create**.

----End

4.10.4.4 Events Supported by Event Monitoring

Table 4-24 Events Supported by Event Monitoring for GeminiDB

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
NoSQL	Instance creation failure	NoSQL CreateInstance Failed	Major	The instance quota or underlying resources are insufficient.	Release the instances that are no longer used and try to provision them again, or submit a service ticket to adjust the quota.	Instances fail to be created.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Specifications change failure	NoSQLResizeInstanceFailed	Major	The underlying resources are insufficient.	Submit a service ticket to ask O&M personnel to coordinate resources, and then try again.	Services are interrupted.
	Node adding failure	NoSQLAddNodesFailed	Major	The underlying resources are insufficient.	Submit a service ticket to ask O&M personnel to coordinate resources, delete the node that failed to be added, and add a new one.	None
	Node deletion failure	NoSQLDeleteNodesFailed	Major	Releasing underlying resources failed.	Delete the node again.	None
	Storage space scale-up failure	NoSQLScaleUpStorageFailed	Major	The underlying resources are insufficient.	Submit a service ticket to ask O&M personnel to coordinate resources, and then try again.	Services may be interrupted.
	Password resetting failure	NoSQLResetPasswordFailed	Major	Resetting the password times out.	Reset the password again.	None
	Parameter template change failure	NoSQLUpdateInstanceParamGroupFailed	Major	Changing a parameter template times out.	Change the parameter template again.	None

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Backup policy configuration failure	NoSQLSetBackupPolicyFailed	Major	The database connection is abnormal.	Configure the backup policy again.	None
	Manual backup creation failure	NoSQLCreateManualBackupFailed	Major	The backup files fail to be exported or uploaded.	Submit a service ticket to O&M personnel.	Data cannot be backed up.
	Automated backup creation failure	NoSQLCreateAutomatedBackupFailed	Major	The backup files fail to be exported or uploaded.	Submit a service ticket to O&M personnel.	Data cannot be backed up.
	Instance status abnormal	NoSQLFaultyDBInstance	Major	This event is a key alarm event and is reported when an instance is faulty due to a disaster or a server failure.	Submit a service ticket.	The database service may be unavailable.
	Instance status recovery	NoSQLDBInstanceRecovered	Major	If a disaster occurs, NoSQL provides an HA tool to automatically or manually rectify the fault. After the fault is rectified, this event is reported.	No further action is required.	None
	Node status abnormal	NoSQLFaultyDBNode	Major	This event is a key alarm event and is reported when a database node is faulty due to a disaster or a server failure.	Check whether the database service is available and submit a service ticket.	The database service may be unavailable.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Node status recovery	NoSQLDBNodeRecovered	Major	If a disaster occurs, NoSQL provides an HA tool to automatically or manually rectify the fault. After the fault is rectified, this event is reported.	No further action is required.	None
	Primary/standby switchover or failover	NoSQLPrimaryStandbySwitched	Major	This event is reported when a primary/secondary switchover or a failover is triggered.	No further action is required.	None
	Occurrence of hotspot partitioning keys	HotKeyOccurs	Major	Hotspot data is stored in one partition because the primary key is improper. Improper application design causes frequent read and write operations on a key.	1. Choose a proper partition key. 2. Add service cache so that service applications read hotspot data from the cache first.	The service request success rate is affected, and the cluster performance and stability deteriorates.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	BigKey occurrence	BigKeyOccurs	Major	The primary key design is improper. There are too many records or too much data in a single partition, causing load imbalance on nodes.	1. Choose a proper partition key. 2. Add a new partition key for hashing data.	As more and more data is stored in the partition, cluster stability deteriorates.
	Insufficient storage space	NoSQLRiskyDataDiskUsage	Major	The storage space is insufficient.	Scale up storage space. For details, see section "Scaling Up Storage Space" in the user guide of GeminiDB.	The instance is set to read-only and data cannot be written to the instance.
	Data disk expanded and being writable	NoSQLDataDiskUsageRecovered	Major	The data disk has been expanded and becomes writable.	No further action is required.	None

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Index creation failure	NoSQL CreateIndexFailed	Major	The service load exceeds what the instance specifications can take. In this case, creating indexes consumes more instance resources. As a result, the response is slow or even frame freezing occurs, and the creation times out.	Select matched instance specifications based on service load. Create indexes during off-peak hours. Create indexes in the background. Select indexes as required.	The index fails to be created or is incomplete. Delete the index and create a new one.
	Write speed decrease	NoSQL Stalling Occurs	Major	The write speed is close to the maximum write speed allowed by the cluster scale and instance specifications. As a result, the database flow control mechanism is triggered, and requests may fail.	1. Adjust the cluster scale or node specifications based on the maximum write rate of services. 2. Measure the maximum write rate of services.	The success rate of service requests is affected.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Data write stopped	NoSQLStoppingOccurs	Major	The data write is too fast, reaching the maximum write capability allowed by the cluster scale and instance specifications. As a result, the database flow control mechanism is triggered, and requests may fail.	1. Change the cluster scale or node specifications based on the maximum write rate of services. 2. Measure the maximum write rate of services.	The success rate of service requests is affected.
	Database restart failure	NoSQLRestartDBFailed	Major	The instance status is abnormal.	Submit a service ticket to O&M personnel.	The instance status may be abnormal.
	Restoration to new instance failure	NoSQLRestoreToNewInstanceFailed	Major	The underlying resources are insufficient.	Submit a service ticket to ask O&M personnel to coordinate resources, and then add new nodes.	Data cannot be restored to a new instance.
	Restoration to existing instance failure	NoSQLRestoreToExistInstanceFailed	Major	The backup file fails to be downloaded or restored.	Submit a service ticket to O&M personnel.	The current instance may be unavailable.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Backup file deletion failure	NoSQLDeleteBackupFailed	Major	The backup files fail to be deleted from OBS.	Delete the backup files again.	None
	Failure to display slow query logs in plaintext	NoSQLSwitchSlowlogPlainTextFailed	Major	The DB API does not support this function.	Refer to the *GeminiDB User Guide* to check whether that the DB API supports the display of slow query logs in plaintext. Submit a service ticket to O&M personnel.	None
	EIP binding failure	NoSQLBindEipFailed	Major	The node status is abnormal, an EIP has been bound to the node, or the EIP to be bound is invalid.	Check whether the node is normal and whether the EIP is valid.	The instance cannot be accessed from a public network.
	EIP unbinding failure	NoSQLUnbindEipFailed	Major	The node status is abnormal or the EIP has been unbound from the node.	Check whether the node and EIP status are normal.	None

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Parameter modification failure	NoSQLModifyParameterFailed	Major	The parameter value is invalid.	Check whether the parameter value is within the valid range and submit a service ticket to O&M personnel.	None
	Parameter template application failure	NoSQLApplyParameterGroupFailed	Major	The instance status is abnormal. So, the parameter template cannot be applied.	Submit a service ticket to O&M personnel.	None
	Enabling or disabling SSL failure	NoSQLSwitchSSLFailed	Major	Enabling or disabling SSL times out.	Try again or submit a service ticket. Do not change the connection mode.	The SSL connection mode cannot be changed.

Event Source	Event Name	Event ID	Event Severity	Description	Solution	Impact
	Too much data in a single row	LargeRowOccurs	Major	If there is too much data in a single row, queries may time out, causing faults like OOM error.	1. Limit the write length of each column and row so that the key and value length of each row does not exceed the preset threshold. 2. Check whether there are abnormal writes or coding, causing large rows.	If there are too many records in a single row, cluster stability will deteriorate as the data volume increases.

4.11 Audit

4.11.1 Key Operations Supported by CTS

With CTS, you can record GeminiDB Mongo key operations for later query, audit, and backtracking.

Table 4-25 GeminiDB Mongo key operations

Operation	Resource Type	Trace Name
Creating an instance	instance	NoSQLCreateInstance
Deleting an instance	instance	NoSQLDeleteInstance
Restarting an instance	instance	NoSQLRestartInstance
Restoring data to a new instance	instance	NoSQLRestoreNewInstance
Scaling up storage space	instance	NoSQLExtendInstanceVolume

Operation	Resource Type	Trace Name
Resetting the password of an instance	instance	NoSQLResetPassword
Changing the name of an instance	instance	NoSQLRenameInstance
Changing specifications	instance	NoSQLResizeInstance
Binding an EIP	instance	NoSQLBindEIP
Unbinding an EIP	instance	NoSQLUnBindEIP
Freezing an instance	instance	NoSQLFreezeInstance
Unfreezing an instance	instance	NoSQLUnfreezeInstance
Creating a backup	backup	NoSQLCreateBackup
Deleting a backup	backup	NoSQLDeleteBackup
Modifying the backup policy of an instance	backup	NoSQLSetBackupPolicy
Adding an instance tag	tag	NoSQLAddTags
Modifying an instance tag	tag	NoSQLModifyInstanceTag
Deleting an instance tag	tag	NoSQLDeleteInstanceTag
Creating a parameter template	parameterGroup	NoSQLCreateConfigurations
Modifying a parameter template	parameterGroup	NoSQLUpdateConfigurations
Modifying instance parameters	parameterGroup	NoSQLUpdateInstanceConfigurations
Replicating a parameter template	parameterGroup	NoSQLCopyConfigurations
Resetting a parameter template	parameterGroup	NoSQLResetConfigurations
Applying a parameter template	parameterGroup	NoSQLApplyConfigurations
Deleting a parameter template	parameterGroup	NoSQLDeleteConfigurations
Enabling SSL	instance	NoSQLSwitchSSL
Changing the security group of an instance	instance	NoSQLModifySecurityGroup
Restoring data to an existing instance	backup	NoSQLRestoreOldInstance

Operation	Resource Type	Trace Name
Exporting parameter template information for an instance	instance	NoSQLSaveConfigurations
Modifying the recycling policy	instance	NoSQLModifyRecyclePolicy

4.11.2 Querying Traces

After CTS is enabled, CTS starts recording operations on cloud resources. The CTS console stores the last seven days of operation records.

This section describes how to query the last seven days of operation records on the CTS console.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click **Service List**. Under **Management & Governance**, click **Cloud Trace Service**.

Step 4 In the navigation pane on the left, click **Trace List**.

Step 5 Specify filter criteria to search for the required traces. The following four filter criteria are available:

- **Trace Source**, **Resource Type**, and **Search By**

  Select filters from the drop-down list.

  When you select **Trace name** for **Search By**, you need to select a specific trace name.

  When you select **Resource ID** for **Search By**, you also need to select or enter a specific resource ID.

  When you select **Resource name** for **Search By**, you also need to select or enter a specific resource name.
- **Operator**: Select a specific operator (a user other than the tenant).
- **Trace Status**: Select **All trace statuses**, **Normal**, **Warning**, or **Incident**.
- **Start Date** and **End Date**: You can specify a time range to query traces.

Step 6 Locate the required trace and click on the left of the trace to view details.

Step 7 Click **View Trace** in the **Operation** column. In the displayed dialog box, the trace structure details are displayed.

----End

4.12 Log Management

4.12.1 Enabling or Disabling Log Reporting

Scenarios

If you enable log reporting for your GeminiDB Mongo instance, new logs generated for the instance will be uploaded to Log Tank Service (LTS).

Precautions

- To use this function, you need to contact customer service to apply for required permissions.
- You will be billed for enabling this function.
- Ensure that there are available LTS log groups and log streams in the same region as your instance.

  For more information about log groups and log streams, see **Log Management**.

Enabling Log Reporting to LTS in Batches

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 In the navigation pane, choose **Log Reporting**.

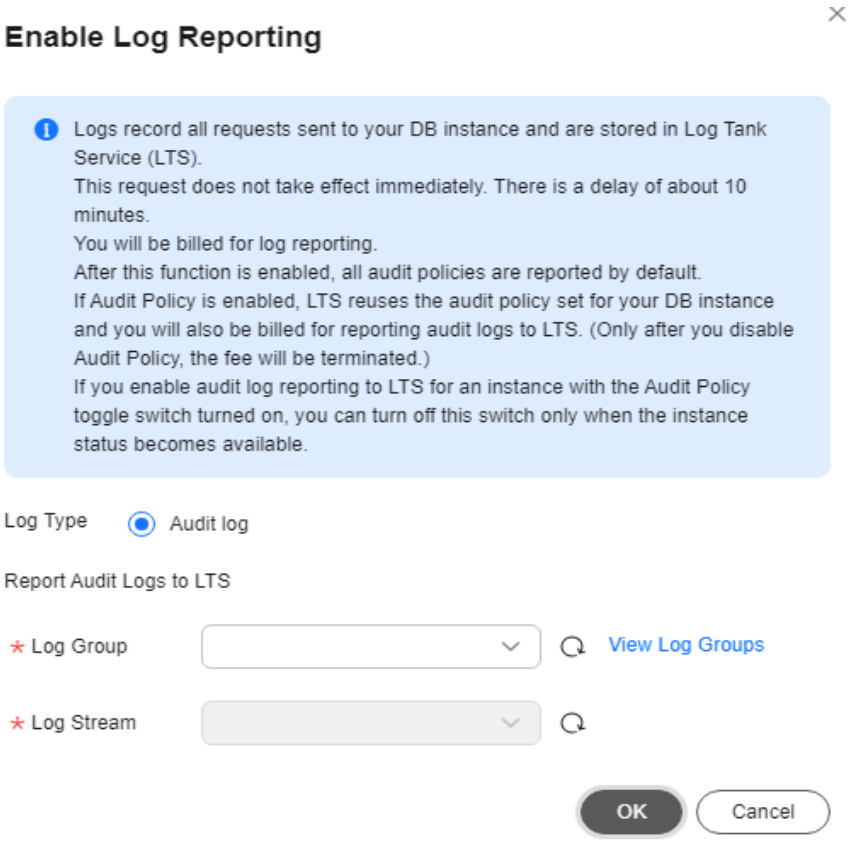
Step 4 Select one or more instances and click **Enable Log Reporting**.

Step 5 Select an LTS log group and log stream and click **OK**.

NOTE

- After this function is enabled, audit logs record all requests sent to your DB instance and are stored in LTS.
- This function does not take effect immediately. There is a delay of about 10 minutes.
- You will be billed for enabling this function. For details, see LTS **pricing details**.
- After this function is enabled, all audit policies are reported by default.
- If Audit Policy is enabled, LTS reuses the audit policy set for your DB instance and you will also be billed for reporting audit logs to LTS. (Only after you disable Audit Policy, the fee will be terminated.)
- If you enable audit log reporting to LTS for an instance with the **Audit Policy** toggle switch turned on, you can turn off this switch only when the instance status becomes available.

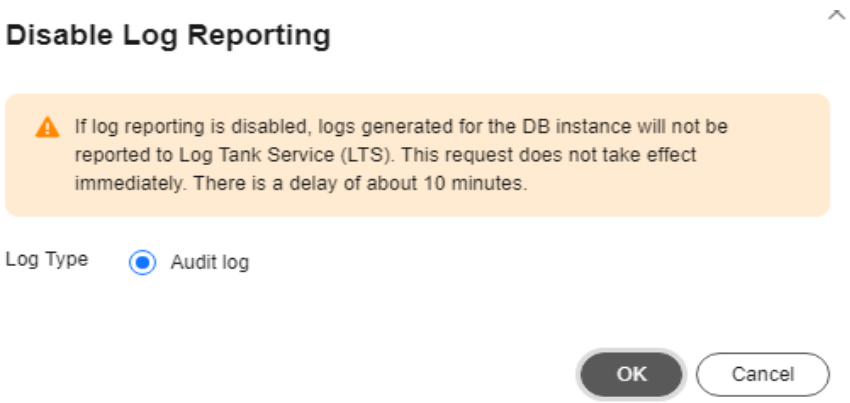
Figure 4-64 Enabling log reporting



Step 6 To disable log reporting, select one or more instances and click **Disable Log Reporting**.

Step 7 In the displayed dialog box, click **OK**.

Figure 4-65 Disabling log reporting



----End

4.12.2 Slow Query Logs

Scenarios

GeminiDB Mongo API allows you to view slow query logs of databases. The unit of the execution time is ms. You can identify the SQL statements that take a long time to execute and tune them based on slow query logs.

Showing Original Logs

NOTE

- After Show Original Log is enabled, it cannot be disabled.
- After Show Original Log is enabled, original logs are displayed. The original slow query logs are displayed for your query and retained for 30 days.
- If the instance to which the slow query log belongs is deleted, related logs are also deleted after Show Original Log is enabled.

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose original logs you want to view and click its name.

Step 4 In the navigation pane on the left, choose **Slow Query Logs**.

Step 5 On the **Slow Query Logs** page, click  .

Step 6 In the displayed dialog box, click **Yes** to enable the function of slowing original logs.

----End

Viewing and Exporting Log Details

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose slow query logs you want to view and click its name.

Step 4 In the navigation pane on the left, choose **Slow Query Logs**.

Step 5 On the **Slow Query Logs** page, set search criteria and click **Search** to view log information.

- Select **All nodes** and view slow query logs of all nodes. Alternatively, select a specific node to view its slow query logs.
- View slow query logs of a specific node in different time ranges.
- View slow query logs of the following types of SQL statements:
  - All SQL statement types
  - SELECT
  - QUERY

  - UPDATE
  - REMOVE
  - GETMORE
  - COMMAND
  - KILLCURSORS

Step 6 On the **Log Details** page, click in the upper right corner of the log list to export log details.

- You can view the exported CSV file to your local PC.
- Up to 2,000 logs can be exported at a time.

----End

4.12.3 Error Logs

Scenarios

GeminiDB Mongo API allows you to view error logs of databases, including warning logs and error logs generated during database running, helping you analyze system issues.

Precautions

This function is in the open beta test (OBT) phase. To use it, contact customer service.

Viewing and Exporting Log Details

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose error logs you want to view and click its name.

Step 4 In the navigation pane on the left, choose **Error Logs**.

Step 5 On the displayed page, click the **Log Details** tab and view log details.

You can view error logs of all nodes or a specific node.

- Select **All nodes** and view error logs of all nodes. Alternatively, select a specific node to view its error logs.
- Select a specific node and view its error logs.
- View error logs of a specific node in different time ranges.
- Choose to view the following levels of error logs:
  - All log levels
  - WARNING
  - ERROR

Step 6 On the **Log Details** page, click in the upper right corner of the log list to export log details.

- You can view the exported CSV file to your local PC.
- Up to 2,000 logs can be exported at a time.

----End

4.13 Billing Management

4.13.1 Renewing Instances

This section describes how to renew your yearly/monthly GeminiDB Mongo instances.

Precautions

Pay-per-use GeminiDB Mongo instances cannot be renewed.

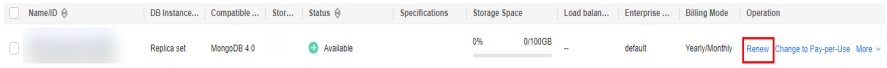
Renewing a Yearly/Monthly Instance

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

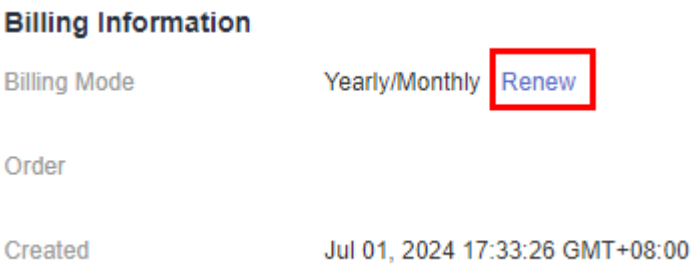
Step 3 On the **Instances** page, locate the instance that you want to renew and click **Renew** in the **Operation** column.

Figure 4-66 Renewal button



Alternatively, click the instance name to go to the **Basic Information** page. In the **Billing Information** area, click **Renew** next to the **Billing Mode** field.

Figure 4-67 Renewal button



Step 4 On the displayed page, renew the instance.

----End

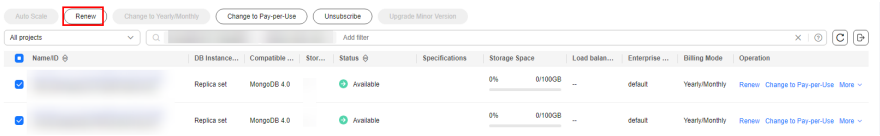
Renewing Instances in Batches

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, select the instances that you want to renew and click **Renew** above the instance list.

Figure 4-68 Renewing instances in batches



Step 4 In the displayed dialog box, click **Yes**.

----End

4.13.2 Changing the Billing Mode from Pay-per-Use to Yearly/Monthly

This section describes how to change the billing mode of a GeminiDB Mongo instance from pay-per-use to yearly/monthly. If you want to use a pay-per-use instance for a long time, change its billing mode to yearly/monthly to reduce costs.

Precautions

Only when the status of a pay-per-use DB instance is **Available**, its billing mode can be changed to yearly/monthly.

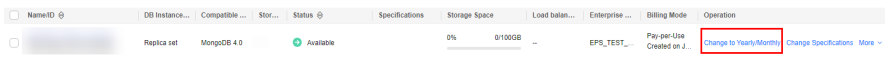
Changing the Billing Mode of a Single Instance

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose billing mode you to change and click **Change to Yearly/Monthly** in the **Operation** column.

Figure 4-69 Changing the billing mode of a single instance



Step 4 On the displayed page, select the renewal duration in month. The minimum duration is one month.

Confirm the settings and click **Pay Now**.

Step 5 Select a payment method and click **Pay**.

Step 6 View the results on the **Instances** page.

In the upper right corner of the instance list, click to refresh the list. The instance status will become **Available** after the change is successful. The billing mode becomes to **Yearly/Monthly**.

----End

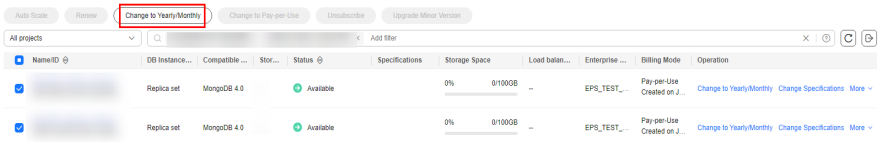
Changing the Billing Mode of Multiple Instance in Batches

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, select the instances whose billing mode you want to change and click **Change to Yearly/Monthly** above the instance list. In displayed dialog box, click **Yes**.

Figure 4-70 Changing the billing mode of multiple instances



Step 4 On the displayed page, select how many months you want to renew the instance for. The minimum duration is one month.

Confirm the settings and click **Pay Now**.

Step 5 Select a payment method and click **Pay**.

Step 6 View the results on the **Instances** page.

In the upper right corner of the instance list, click to refresh the list. The instance status will become **Available** after the change is successful. The billing mode changes to **Yearly/Monthly**.

----End

4.13.3 Changing the Billing Mode from Yearly/Monthly to Pay-per-Use

You can change the billing mode of a GeminiDB Mongo instance from yearly/monthly to pay-per-use and then pay only for the actual usage of your resources.

Precautions

The billing mode of a yearly/monthly instance can only be changed to pay-per-use when the instance is in the **Available** status.

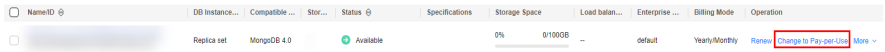
Changing the Billing Mode of a Single Instance to Pay-per-Use

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance whose billing mode you want to change and click **Change to Pay-per-Use** in the **Operation** column.

Figure 4-71 Changing from yearly/monthly to pay-per-use



Step 4 On the displayed page, confirm the instance information and click **Change to Pay-per-Use**. The billing mode will change to pay-per-use after the instance expires.

NOTICE

Auto renewal will be disabled after the billing mode of your instances change to pay-per-use. Exercise caution when performing this operation.

Step 5 After you submit the change, check whether a message is displayed in the **Billing Mode** column, indicating that the billing mode will be changed to pay-per-use after the instance expires.

Step 6 To cancel the change, choose **Billing** > **Renewal** to enter the Billing Center. On the **Renewals** page, locate the instance and click **More** > **Cancel Change to Pay-per-Use**.

Step 7 In the displayed dialog box, click **Yes**.

----End

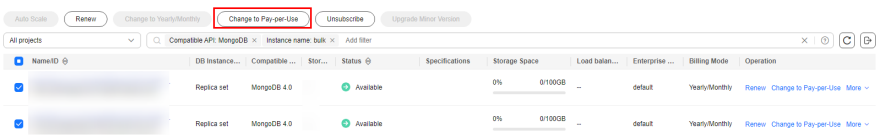
Changing the Billing Mode of Multiple Instances to Pay-per-use

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, select the instances whose billing mode you want to change and click **Change to Pay-per-Use** above the instance list.

Figure 4-72 Changing the billing mode of multiple instances to pay-per-use



Step 4 In the displayed dialog box, click **Yes**.

Step 5 On the displayed page, confirm the instance information and click **Change to Pay-per-Use**. The billing mode will change to pay-per-use after the instance expires.

NOTICE

Auto renewal will be disabled after the billing mode of your instances change to pay-per-use. Exercise caution when performing this operation.

Step 6 After you submit the change, check whether a message is displayed in the **Billing Mode** column, indicating that the billing mode will be changed to pay-per-use after the instance expires.

Step 7 To cancel the change, choose **Billing** > **Renewal** to enter the Billing Center. On the **Renewals** page, locate the target instance and click **More** > **Cancel Change to Pay-per-Use**.

Step 8 In the displayed dialog box, click **Yes**.

----End

4.13.4 Unsubscribing from a Yearly/Monthly DB Instance

If you do not need a yearly/monthly instance any longer, unsubscribe from it.

Precautions

- Unsubscribing operations cannot be undone. Exercise caution when performing this operation. To retain data, create a manual backup before unsubscription. For details, see **Creating a Manual Backup**.
- After an unsubscription request is submitted, resources and data will be deleted and cannot be retrieved. Ensure that the manual backup is complete before submitting the unsubscription request.

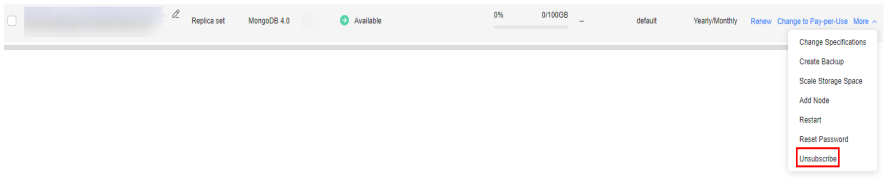
Unsubscribing from a Single Yearly/Monthly Instance

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, locate the instance you want to unsubscribe from and choose **More** > **Unsubscribe** in the **Operation** column.

Figure 4-73 Unsubscribing from a yearly/monthly instance



Step 4 In the displayed dialog box, click **Yes**.

Step 5 On the displayed page, confirm the order to be unsubscribed and select a reason. Then, click **Confirm**.

For details, see **Unsubscription Rules**.

Step 6 In the displayed dialog box, click **Yes**.

NOTICE

1. After an unsubscription request is submitted, resources and data will be deleted and cannot be retrieved.
2. If you want to retain data, complete a manual backup before submitting the unsubscription request.

Step 7 View the unsubscription result. After the instance order is successfully unsubscribed, the instance is no longer displayed in the instance list on the **Instances** page.

----End

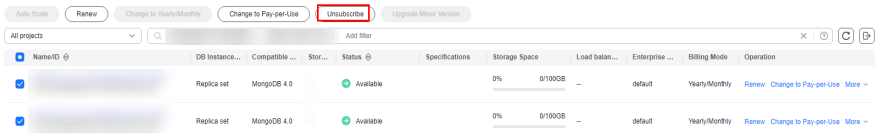
Unsubscribing from Multiple Yearly/Monthly Instances

Step 1 Log in to the management console.

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 Choose **Instances** in the navigation pane on the left, select the instances you want to unsubscribe from and click **Unsubscribe** above the instance list.

Figure 4-74 Unsubscribing from multiple yearly/monthly instances



Step 4 In the displayed dialog box, click **Yes**.

Step 5 On the displayed page, confirm the order to be unsubscribed and select a reason. Then, click **Confirm**.

For details, see **Unsubscription Rules**.

Step 6 In the displayed dialog box, click **Yes**.

NOTICE

1. After an unsubscription request is submitted, resources and data will be deleted and cannot be retrieved.
2. If you want to retain data, complete a manual backup before submitting the unsubscription request.

Step 7 View the unsubscription result. After the instance order is successfully unsubscribed, the instance is no longer displayed in the instance list on the **Instances** page.

----End

4.14 Quotas

Scenarios

Quotas are enforced for service resources on the platform to prevent unforeseen spikes in resource usage. Quotas limit the number or amount of resources available to users, for example, the maximum number of GeminiDB instances that you can create.

If a quota cannot meet your needs, apply for a higher quota.

Viewing Quotas

Step 1 Log in to the management console.

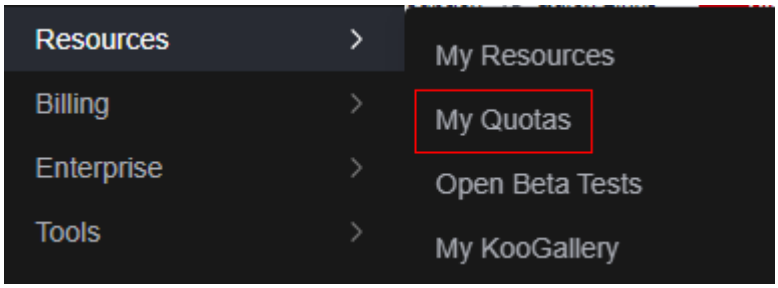
Step 2 In the service list, choose **Databases** > **GeminiDB Redis API**.

Step 3 Click in the upper left corner and select a region and project.

Step 4 In the upper right corner, choose **Resources** > **My Quotas**.

The **Quota** page is displayed.

Figure 4-75 My quotas



Step 5 On the **Quotas** page, view the used and total quotas of each type of GeminiDB resources.

----End

Increasing Quotas

Step 1 Log in to the management console.

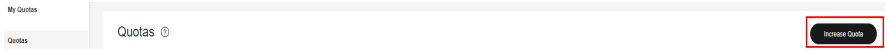
Step 2 In the service list, choose **Databases** > **GeminiDB Redis API**.

Step 3 Click in the upper left corner and select a region and project.

Step 4 In the upper right corner, choose **Resources** > **My Quotas**.

Step 5 In the upper right corner of the page, click **Increase Quota**.

Figure 4-76 Increasing quotas



Step 6 On the **Create Service Ticket** page, configure parameters as required.

In the **Problem Description** area, enter the required quota and reason for the quota adjustment.

Step 7 After all necessary parameters are configured, select the agreement and click **Submit**.

----End

5 Best Practices

5.1 Buying and Connecting to a GeminiDB Mongo Instance

This section describes how to buy a GeminiDB Mongo instance and uses a Linux ECS as an example to describe how to connect to the instance over a private network.

- **Step 1: Buy a GeminiDB Mongo Instance**
- **Step 2: Buy an Instance**
- **Step 3: Connect to the GeminiDB Mongo Instance**

Step 1: Buy a GeminiDB Mongo Instance

Step 1 **Log in to the management console.**

Step 2 In the service list, choose **Databases** > **GeminiDB Mongo API**.

Step 3 On the **Instances** page, click **Buy DB Instance**.

Step 4 Click **Buy DB Instance**, select a billing mode, and configure instance parameters. Then, click **Next** and complete subsequent operations.

Figure 5-1 Basic information

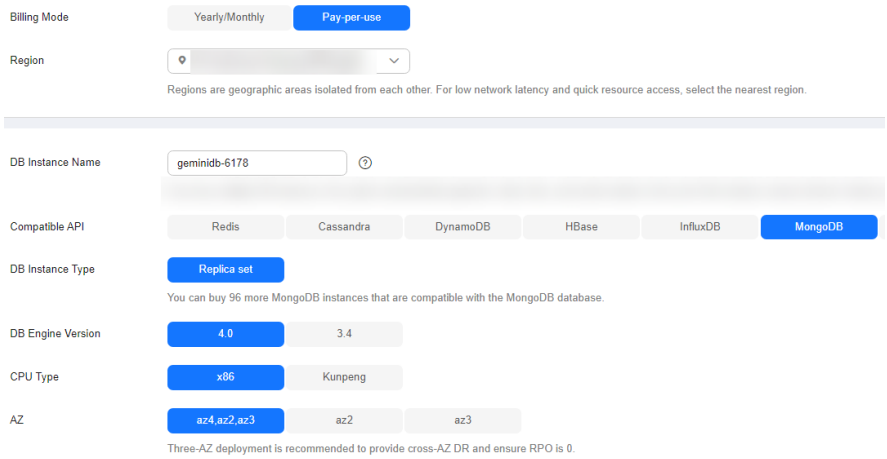


Figure 5-2 Selecting specifications

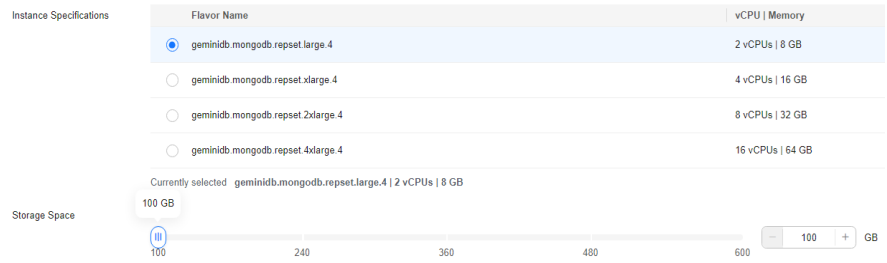
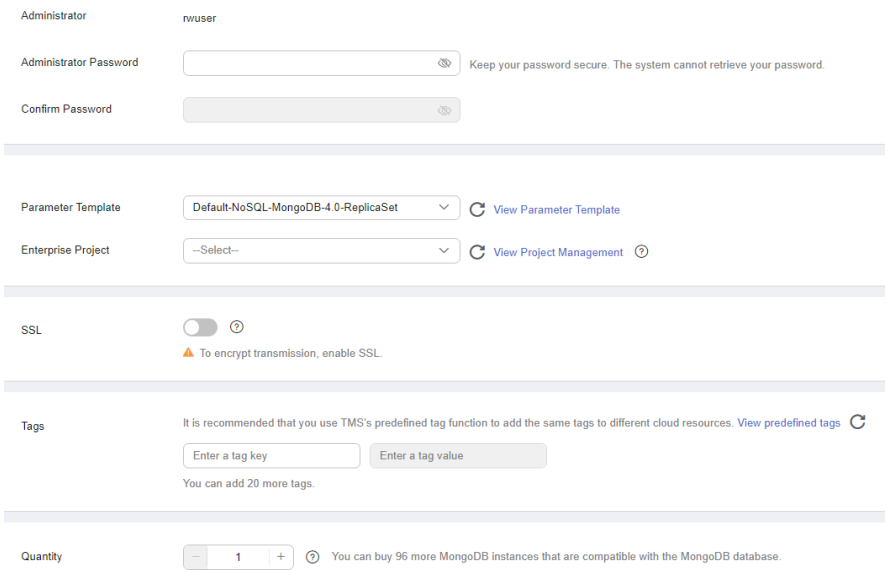


Figure 5-3 Setting a password



Step 5 View the purchased GeminiDB Mongo instance.

Figure 5-4 Successful purchase

----End

Step 2: Buy an Instance

Step 1 **Log in to the management console.**

Step 2 In the service list, choose **Compute** > **Elastic Cloud Server**. On the Elastic Cloud Server console, click **Buy ECS**.

Step 3 Configure basic settings and click **Next: Configure Network**. Make sure that the ECS is in the same region, AZ, VPC, and security group as the GeminiDB Mongo instance you created.

Figure 5-5 Basic settings

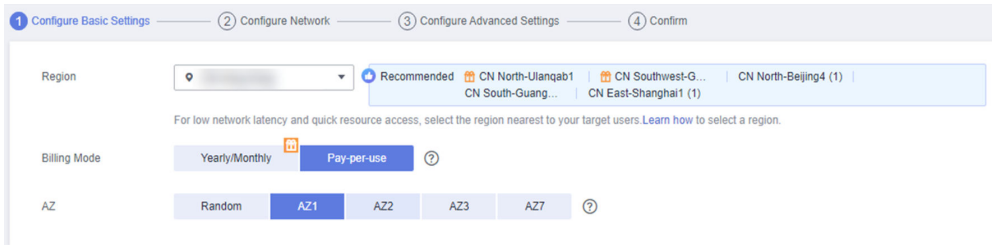


Figure 5-6 Selecting specifications

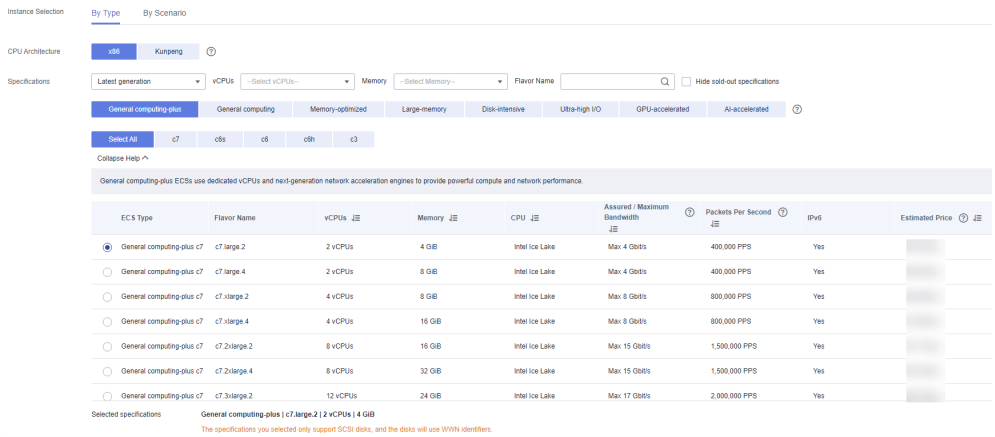
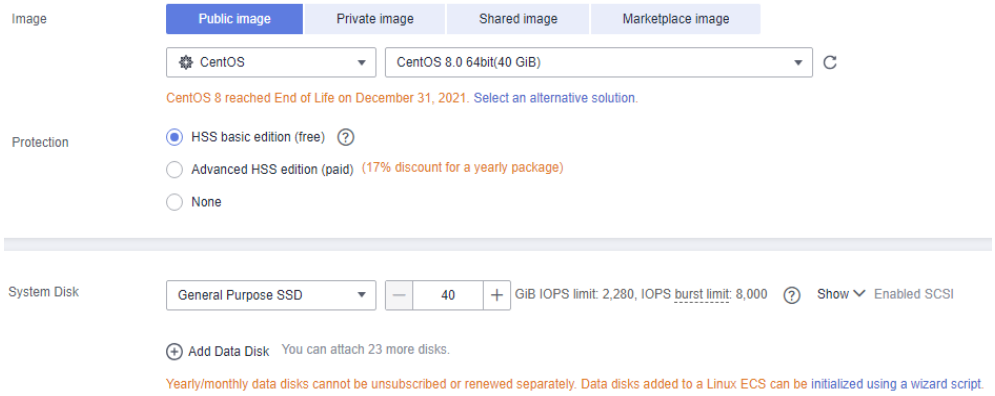


Figure 5-7 Selecting an image



Step 4 Configure the ECS network and click **Next: Configure Advanced Settings**. Make sure that the ECS is in the same VPC and security group as the GeminiDB Mongo instance.

- If security group rules allow access from the ECS, you can connect to the instance using the ECS.
- If the security group rules do not allow access from the ECS, add an inbound rule to the security group.

Figure 5-8 Network settings

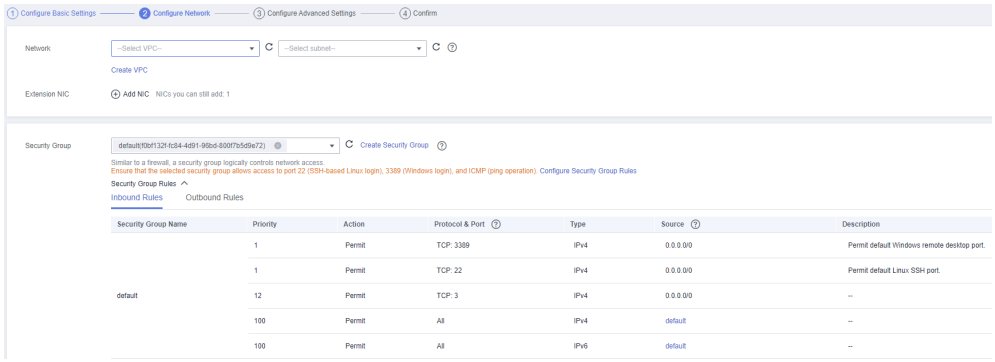
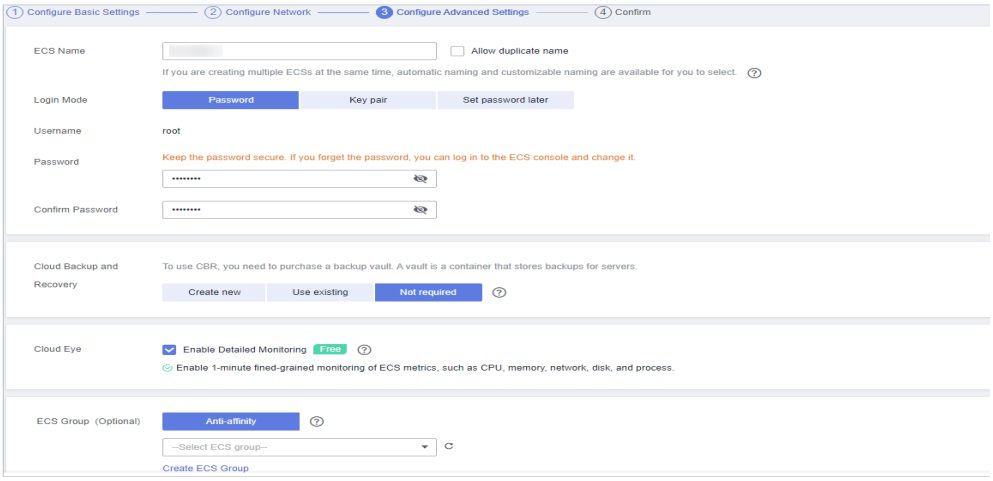


Figure 5-9 Selecting an EIP

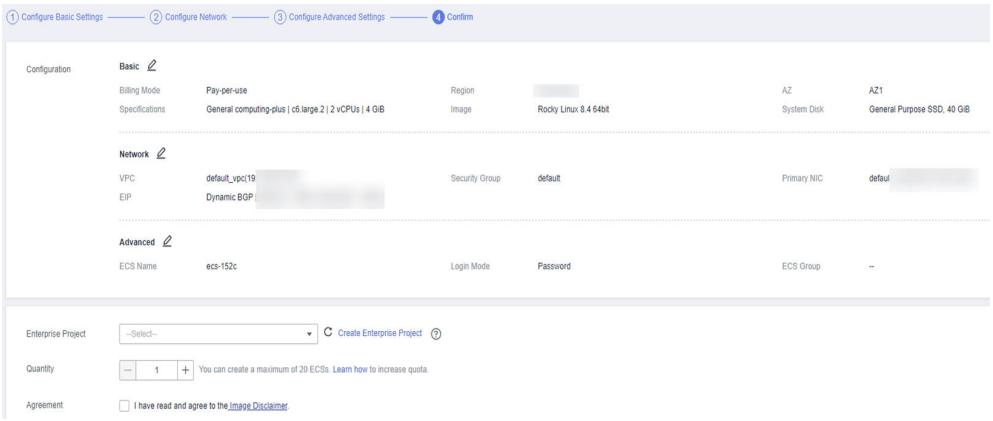
Step 5 Configure a password for the ECS and click **Next: Confirm**.

Figure 5-10 Advanced settings



Step 6 Confirm the configurations and click **Submit**.

Figure 5-11 Confirming the configurations



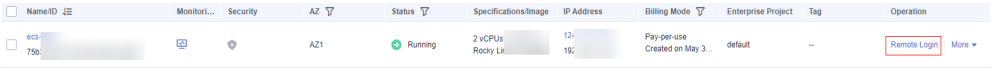
Step 7 View the purchased ECS.

----End

Step 3: Connect to the GeminiDB Mongo Instance

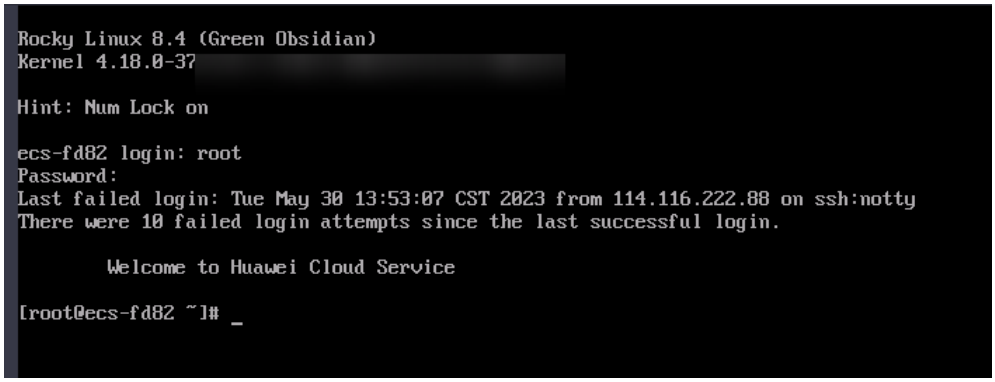
Step 1 On the ECS console, log in to the ECS using the remote login option.

Figure 5-12 Remote login



Step 2 Enter the username and password of the ECS.

Figure 5-13 Entering the username and password



Step 3 Install the MongoDB client by referring to **How Can I Install a MongoDB Client?**

Step 4 Connect to the instance in the directory where the MongoDB client is located.

- **Connect to a replica set using its connection address (recommended).**

  GeminiDB Mongo provides a connection address for each instance on the console. The address includes IP addresses of the primary, standby and hidden nodes and their port numbers. Connecting to a replica set with this method allows access to the primary, standby, and hidden nodes at the same time, so write errors can be avoided after a primary/standby switchover.

  Command format:

  ./mongo "*<Connection address>*"

  The following is an example command:

  ./mongo "mongodb://
  rwuser:**@192.168.0.196:8635,192.168.0.67:8635,192.168.0.32:8635/test?**
  authSource=admin&replicaSet=replica"

Table 5-1 Required description

Parameter	Description
<Connection address>	Connection address of the replica set. You can obtain the connection address in either of the following ways: In the instance list, locate the replica set and click its name. On the **Basic Information** page, in the **Network Information** area, view the connection address beside the **Address** field. Alternatively, click the instance name to go to the **Basic Information** page. In the navigation pane on the left, choose **Connections** to obtain the address of the instance. – In the preceding command, ***** needs to be replaced with the password of the instance administrator. If the password contains at signs (@),exclamation marks (!), or percent signs (%), replace them with hexadecimal URL codes %40, %21, and %25 respectively. For example, if the password is ****@%***!, enter URL code is ****%40%25***%21. – **replica** in **replicaSet=replica** indicates the name of a replica set. All Huawei Cloud GeminiDB Mongo replica sets are named **replica** here and cannot be changed.

If information similar to the following is displayed, the connection was successful.

```
replica:PRIMARY>
```

- **Connect to a node of the replica set.**

  You can also access a specified node of the replica set using the node private IP address.

  Command format:

  ./mongo --host <*DB_HOST*> **--port** <*DB_PORT*> **-u** <*DB_USER*> **-p --authenticationDatabase admin**

  Example:

  ./mongo --host 192.168.1.6 --port 8635 -u rwuser -p --authenticationDatabase admin

Table 5-2 Required description

Parameter	Description
<*DB_HOST*>	Private IP address of the primary or standby node of the replica set to be connected. – If the primary node is accessed, you can read and write data from and to the replica set. – If the standby node is accessed, you can only read data from the replica set. You can locate the replica set and click its name to switch to the **Basic Information** page, and then view the private IP address of the required replica set node in the node list. Alternatively, click the instance name to go to the **Basic Information** page. In the navigation pane on the left, choose **Connections** and view the private IP address of the node.
<*DB_PORT*>	Database port.
<*DB_USER*>	Username of the instance administrator. The default value is **rwuser**.

Enter the password of the database account if the following information is displayed:

```
Enter password:
```

If information similar to the following is displayed, the connection was successful.

Connection results if the primary node is connected:

```
replica:PRIMARY>
```

Connection results if the standby node is connected:

```
replica:SECONDARY>
```

----End

5.2 Connecting to an Instance Using Java

5.2.1 Driver Package and Environment Dependency

GeminiDB Mongo allows you to perform operations using Java. In this way, you can connect to an instance through an SSL connection or an unencrypted connection. The SSL connection encrypts data and is more secure.

SSL is disabled by default for a new GeminiDB Mongo instance. To enable SSL, see **Enabling SSL**.

Installing the Driver

1. Click the **JAR driver download address** to download **mongo-java-driver-3.12.9.jar**, which provides APIs for accessing GeminiDB Mongo instances.
2. To install the driver, see the **official guide**.

Environment

JDK 1.8 must be configured for the client. JDK is cross-platform and supports Windows and Linux.

The following uses Windows as an example to describe how to configure JDK:

Step 1 In the DOS window, run **java -version** to check the JDK version. Ensure that the JDK version is 1.8. If JDK is not installed, download the installation package and install it.

Step 2 Right-click the **This PC** icon on the desktop of the Windows OS and choose **Properties** from the shortcut menu.

Step 3 In the **System** window, click **Advanced system settings** in the navigation tree on the left.

Step 4 In the dialog box that is displayed, click **Environment Variables**.

Step 5 In the displayed window, set the variables listed in the following table.

Variable	Operation	Value
JAVA_HOME	• If this parameter exists, click **Edit**. • If this parameter does not exist, click **New**.	Java installation directory. For example, C:\Program Files\Java\jdk1.8.0_131.
Path	Click **Edit**.	• If JAVA_HOME is configured, add **%JAVA_HOME %\bin** to the beginning of the variable value. • If JAVA_HOME is not configured, add the following Java installation path before the variable value. C:\Program Files\Java\jdk1.8.0_131\bin;
CLASSPATH	Click **New**.	.;%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar;

Step 6 Click **OK** and close the windows in sequence.

----End

5.2.2 Connecting to a Database

Connecting to an Instance Using an SSL Certificate

NOTE

Download the SSL certificate and verify the certificate before connecting to an instance.

In the **DB Information** area on the **Basic Information** page, click in the **SSL** field to download the root certificate or certificate bundle.

Step 1 Use Java to connect to the MongoDB database.

- Connecting to a replica set
  mongodb://<username>**:**<password>**@**<instance_ip>**:**<instance_port>**/**<database_name>**?authSource=admin&replicaSet=replica&ssl=true**

Table 5-3 Parameters

Parameter	Description
<username>	Current username.
<password>	Password for the current username.
<instance_ip>	If you access an instance from an ECS, *instance_ip* is the private IP address of the instance.
	If you access an instance through an EIP, *instance_ip* is the EIP that has been bound to the instance.
<instance_port>	Database port displayed on the **Basic Information** page. Default value: **8635**
<database_name>	Name of the database to be connected.
authSource	Authentication user database. The value is **admin**.
ssl	Connection mode. **true** indicates that SSL will be used.

Example script in Java:

```
import java.util.ArrayList;
import java.util.List;
import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;
import com.mongodb.ServerAddress;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.MongoCollection;
import com.mongodb.MongoClientURI;
import com.mongodb.MongoClientOptions;
public class MongoDBJDBC {
public static void main(String[] args){
    try {
            System.setProperty("javax.net.ssl.trustStore", "/home/Mike/jdk1.8.0_112/jre/lib/
security/mongostore");
            System.setProperty("javax.net.ssl.trustStorePassword", "****");
```



```
            ServerAddress serverAddress = new ServerAddress("ip", port);
            List addrs = new ArrayList();
            addrs.add(serverAddress);
            // There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
            // In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
            String userName = System.getenv("EXAMPLE_USERNAME_ENV");
            String rwuserPassword = System.getenv("EXAMPLE_PASSWORD_ENV");
            MongoCredential credential = MongoCredential.createScramSha1Credential("rwuser", "admin", rwuserPassword.toCharArray());
            List credentials = new ArrayList();
            credentials.add(credential);
            MongoClientOptions opts= MongoClientOptions.builder()
            .sslEnabled(true)
            .sslInvalidHostNameAllowed(true)
            .build();
            MongoClient mongoClient = new MongoClient(addrs,credentials,opts);
            MongoDatabase mongoDatabase = mongoClient.getDatabase("testdb");
            MongoCollection collection = mongoDatabase.getCollection("testCollection");
            Document document = new Document("title", "MongoDB").
            append("description", "database").
            append("likes", 100).
            append("by", "Fly");
            List documents = new ArrayList();
            documents.add(document);
            collection.insertMany(documents);
            System.out.println("Connect to database successfully");
            } catch (Exception e) {
            System.err.println( e.getClass().getName() + ": " + e.getMessage() );
        }
    }
}
```

Sample code:

```
javac -cp .:mongo-java-driver-3.2.0.jar MongoDBJDBC.java
java -cp .:mongo-java-driver-3.2.0.jar MongoDBJDBC
```

----End

Connecting to an Instance Without the SSL Certificate

NOTE

You do not need to download the SSL certificate because certificate verification on the server is not required.

Step 1 Use Java to connect to the MongoDB database.

- Connecting to a replica set

  mongodb://<username>**:**<password>**@**<instance_ip>**:**<instance_port>**/**<database_name>**?authSource=admin&replicaSet=replica**

  Table 5-4 Parameters

Parameter	Description
<username>	Current username.

Parameter	Description
<password>	Password for the current username.
<instance_ip>	If you access an instance from an ECS, *instance_ip* is the private IP address of the instance.
	If you access an instance through an EIP, *instance_ip* is the EIP that has been bound to the instance.
<instance_port>	Database port displayed on the **Basic Information** page. Default value: **8635**
<database_name>	Name of the database to be connected.
authSource	Authentication user database. The value is **admin**.

Example script in Java:

```
import com.mongodb.ConnectionString;
import com.mongodb.reactivestreams.client.MongoClients;
import com.mongodb.reactivestreams.client.MongoClient;
import com.mongodb.reactivestreams.client.MongoDatabase;
import com.mongodb.MongoClientSettings;
public class MyConnTest {
    final public static void main(String[] args) {
    try {
        // no ssl
    ConnectionString connString = new ConnectionString("mongodb://
rwuser:****@192.*.*.*:8635,192.*.*.*:8635/test? authSource=admin");
    MongoClientSettings settings = MongoClientSettings.builder()
        .applyConnectionString(connString)
        .retryWrites(true)
        .build();
    MongoClient mongoClient = MongoClients.create(settings);
    MongoDatabase database = mongoClient.getDatabase("test");
     System.out.println("Connect to database successfully");
    } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Test failed");
        }
}
}
```

----End

Connecting to an Instance Using YML file

Step 1 Use YML file to connect to the MongoDB database.

- Connecting to a replica set

```
spring:
  data:
    mongodb:
      uri: mongodb://<username>:<password>@<instance_ip>:<instance_port>/<database_name>?
authSource=admin&replicaSet=replica
```


Table 5-5 Parameters

Parameter	Description
<username>	Current username.
<password>	Password for the current username.
<instance_ip>	If you access an instance from an ECS, *instance_ip* is the private IP address of the instance.
	If you access an instance through an EIP, *instance_ip* is the EIP that has been bound to the instance.
<instance_port>	Database port displayed on the **Basic Information** page. Default value: **8635**
<database_name>	Name of the database to be connected.
authSource	Authentication user database. The value is **admin**.

Example script using YML file:

- application.yml

```
spring:
  data:
    mongodb:
      uri: mongodb://<username>:<password>@<instance_ip>:<instance_port>/
<database_name>?authSource=admin&replicaSet=replica
```

- ITestService.java

```
package com.springboot.cli.service;
public interface ITestService {
    long doCount();
}
```

- TestServiceImpl.java

```
package com.springboot.cli.service.impl;
import com.springboot.cli.service.ITestService;
import lombok.RequiredArgsConstructor;
import org.springframework.data.mongodb.core.MongoTemplate;
import org.springframework.stereotype.Service;
@Service
@RequiredArgsConstructor
public class TestServiceImpl implements ITestService {
    private final MongoTemplate mongoTemplate;
    @Override
    public long doCount() {
        return mongoTemplate.estimatedCount("coll");
    }
}
```

- MongodbApplicationTests.java

```
package com.springboot.cli;
import com.springboot.cli.service.impl.TestServiceImpl;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.stereotype.Service;
@SpringBootTest
```



```
@Service
class MongodbApplicationTests {
    @Autowired
    TestServiceImpl testService;
    @Test
    void doCount() {
        System.out.println("coll count is: " + testService.doCount());
    }
}
```

----End

5.2.3 Accessing a Database

Before accessing a database, import the following classes:

```
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import static com.mongodb.client.model.Filters.*;
import com.mongodb.client.model.CreateCollectionOptions;
import com.mongodb.client.model.ValidationOptions;
```

Accessing DataBase

If an initialized MongoClient instance exists, run the following command to access a database:

```
MongoDatabase database = mongoClient.getDatabase("test");
```

Assessing a Collection

After obtaining a MongoDatabase instance, run the following command to obtain a collection:

```
MongoCollection<Document> coll = database.getCollection("testCollection");
```

Creating a Collection

You can use the createCollection() method to create a collection and specify the attributes of the collection.

```
database.createCollection("testCollection", new CreateCollectionOptions()..sizeInBytes(200000))
```

Inserting Data

```
Document doc0 = new Document("name", "zhangsan")
            .append("age", 3)
            .append("sex", "male");

Document doc1 = new Document("name", "LiSi")
            .append("age", 2)
            .append("sex", "female");

Document doc2 = new Document("name", "wangmazi")
            .append("age", 5)
            .append("sex", "male");

List<Document> documents = new ArrayList<Document>();
```



```
documents.add(doc1);
documents.add(doc2);

collection.insertMany(documents);
```

Deleting Data

```
collection.deleteOne(eq("_id", new ObjectId("00000001")));
```

Deleting a Table

```
MongoCollection<Document> collection = database.getCollection("test");
collection.drop()
```

Reading Data

```
MongoCollection<Document> collection = database.getCollection("contacts");
MongoCursor<String> cursor = collection.find();
while (cursor.hasNext())  {
   Object result = cursor.next();
}
```

Query with Filter Criteria

```
MongoCollection<Document> collection = database.getCollection("test");
MongoCursor<String> cursor = collection.find(
    new Document("name","zhangsan")
        .append("age: 5));
while (cursor.hasNext())  {
   Object result = cursor.next();
}
```

Running Commands

Run **buildInfo** and **collStats**.

```
MongoClient mongoClient = MongoClients.create();
MongoDatabase database = mongoClient.getDatabase("test");

Document buildInfoResults = database.runCommand(new Document("buildInfo", 1));
System.out.println(buildInfoResults.toJson());

Document collStatsResults = database.runCommand(new Document("collStats", "restaurants"));
System.out.println(collStatsResults.toJson());
```

Creating an Index

```
MongoClient mongoClient = MongoClients.create();
MongoDatabase database = mongoClient.getDatabase("test");
MongoCollection<Document> collection = database.getCollection("test");
collection.createIndex(Indexes.ascending("age"));
```

5.2.4 Complete Example

```
package mongodbdemo;
import org.bson.*;
import com.mongodb.*;
import com.mongodb.client.*;
public class MongodbDemo {
    public static void main(String[] args) {
```



```
    // There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
    // In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
    String userName = System.getenv("EXAMPLE_USERNAME_ENV");
    String rwuserPassword = System.getenv("EXAMPLE_PASSWORD_ENV");
    String mongoUri = "mongodb://" + userName + ":" + rwuserPassword + "@10.66.187.127:27017/admin";
    MongoClientURI connStr = new MongoClientURI(mongoUri);
    MongoClient mongoClient = new MongoClient(connStr);
    try {
        //Use the database named someonedb.
        MongoDatabase database = mongoClient.getDatabase("someonedb");
        //Obtain the someonetable handle of the collection/table.
        MongoCollection<Document> collection = database.getCollection("someonetable");
         //Prepare data to be written.
        Document doc = new Document();
        doc.append("key", "value");
        doc.append("username", "jack");
        doc.append("age", 31);
         //Write data.
        collection.insertOne(doc);
        System.out.println("insert document: " + doc);
         //Read data.
        BsonDocument filter = new BsonDocument();
        filter.append("username", new BsonString("jack"));
        MongoCursor<Document> cursor = collection.find(filter).iterator();
        while (cursor.hasNext()) {
            System.out.println("find document: " + cursor.next());
        }
    } finally {
        //Close the connection.
        mongoClient.close();
    }
  }
}
```

For more information about Java APIs, see the **official documents**.

5.3 Connecting to an Instance Using Python

5.3.1 PyMongo Package

Python uses PyMongo to provide a unified API for accessing GeminiDB Mongo databases. Applications can perform operations using PyMongo. PyMongo supports SSL connections and uses a connection pool to support multithreaded applications.

To install PyMongo, see the **official guide**.

5.3.2 Connecting to a Database

If you are connecting to an instance using Python, an SSL certificate is optional, but downloading an SSL certificate and encrypting the connection will improve the security of your instance.

SSL is disabled by default for a new GeminiDB Mongo instance. To enable SSL, see **Enabling SSL**.

Prerequisites

1. To connect an ECS to an instance, the ECS must be able to communicate with the GeminiDB Mongo instance. You can run the following command to connect to the IP address and port of the instance server to test the network connectivity.

   curl *ip:port*

   If the message **It looks like you are trying to access MongoDB over HTTP on the native driver port** is displayed, the ECS and DDS instance can communicate with each other.
2. Install Python and third-party installation package **pymongo** on the ECS. PyMongo 3.10 is recommended.
3. If SSL is enabled, download the root certificate and upload it to the ECS.

Connection Code

- Enabling SSL

```
import ssl
import os
from pymongo import MongoClient
# There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
# In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
rwuser = os.getenv('EXAMPLE_USERNAME_ENV')
password = os.getenv('EXAMPLE_PASSWORD_ENV')
conn_urls="mongodb://%s:%s@ip:port/{mydb}?authSource=admin"
connection = MongoClient(conn_urls % (rwuser,
password),connectTimeoutMS=5000,ssl=True,
ssl_cert_reqs=ssl.CERT_REQUIRED,ssl_match_hostname=False,ssl_ca_certs=${path to certificate authority file})
dbs = connection.database_names()
print "connect database success! database names is %s" % dbs
```

- Disabling SSL

```
import ssl
import os
from pymongo import MongoClient
# There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
# In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
rwuser = os.getenv('EXAMPLE_USERNAME_ENV')
password = os.getenv('EXAMPLE_PASSWORD_ENV')
conn_urls="mongodb://%s:%s@ip:port/{mydb}?authSource=admin"
connection = MongoClient(conn_urls % (rwuser, password),connectTimeoutMS=5000)
dbs = connection.database_names()
print "connect database success! database names is %s" % dbs
```

NOTE

The authentication database in the URL must be **admin**. Set **authSource** to **admin**.

5.3.3 Accessing a Database

Assume that the client application has connected to the database and a MongoClient client is initialized.

Accessing DataBase

If an initialized MongoClient instance exists, run the following command to access a database:

```
db=client.test_database
```

Alternatively, use the following method:

```
db=client["test_database"]
```

Assessing a Collection

```
collection=db.test_collection
```

Alternatively, use the following method:

```
collection=db["test_collection"]
```

Creating a Collection

You can use the createCollection() method to create a collection and specify the attributes of the collection.

```
collection = db.create_collection("test")
```

Inserting Data

```
student = {
    'id': '20170101',
    'name': 'Jordan',
    'age': 20,
    'gender': 'male'
}
result = collection.insert(student);
```

Deleting Data

```
result = collection.delete_one({'name': 'Kevin'})
```

Deleting a Table

```
db.drop_collection("test")
```

Reading Data

```
result = collection.find_one({'name': 'Mike'})
```

Query with Filter Criteria

```
result = collection.find_one({"author":"Mike"}
```


Running Commands

Run **buildInfo** and **collStats**.

```
db.command("collstats","test")
db.command("buildinfo")
```

Counting

```
count = collection.find().count()db.command("buildinfo")
```

Sorting

```
results = collection.find().sort('name', pymongo.ASCENDING)
print([result['name'] for result in results])
```

Creating an Index

```
result=db.profiles.create_index([('user_id',pymongo.ASCENDING)],... unique=True)
```

5.3.4 Complete Example

```
#!/usr/bin/python
import pymongo
import random
import os
# There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
# In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
username = os.getenv('EXAMPLE_USERNAME_ENV')
password = os.getenv('EXAMPLE_PASSWORD_ENV')
mongodbUri = 'mongodb://%s:%s@10.66.187.127:27017/admin'
client = pymongo.MongoClient(mongodbUri % (username, password ))
db = client.somedb
db.user.drop()
element_num=10
for id in range(element_num):
  name = random.choice(['R9','cat','owen','lee','J'])
  sex = random.choice(['male','female'])
  db.user.insert_one({'id':id, 'name':name, 'sex':sex})
content = db.user.find()
for i in content:
  print i
```

For more information about PyMongo APIs, see the **official document**.

5.4 Connecting to an Instance Using Golang

5.4.1 Driver Package

GeminiDB Mongo allows you to operate data using Go. You can connect to an instance through an SSL connection or an unencrypted connection. SSL connection is more secure.

SSL is disabled by default for a new GeminiDB Mongo instance. To enable SSL, see **Enabling SSL**.

Downloading the Driver

To download the driver, go.mod is recommended.

```
require go.mongodb.org/mongo-driver v1.7.6
```

Import the go files.

```
import (
"go.mongodb.org/mongo-driver/bson"
"go.mongodb.org/mongo-driver/mongo"
"go.mongodb.org/mongo-driver/mongo/options"
"go.mongodb.org/mongo-driver/mongo/readpref"
)
```

5.4.2 Connecting to a Database

Prerequisites

1. To connect an ECS to an instance, the ECS must be able to communicate with the GeminiDB Mongo instance. You can run the following command to connect to the IP address and port of the instance server to test the network connectivity.

   curl *ip:port*

   If the message **It looks like you are trying to access MongoDB over HTTP on the native driver port** is displayed, the ECS and GeminiDB Mongo instance can communicate with each other.
2. If SSL is enabled, download the root certificate and upload it to the ECS.

Connection Code

- Enabling SSL

```
//Construct an authentication credential.
// There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
// In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
username = System.getenv("EXAMPLE_USERNAME_ENV")
password = System.getenv("EXAMPLE_PASSWORD_ENV")
credential := options.Credential{
  AuthMechanism: "SCRAM-SHA-1",
  AuthSource:    "admin",
  Username:      username,
  Password:      password,
}
//HA URI. Note that SetDirect is set to false.
highProxyUri := "mongodb://host1:8635,host2:8635/?ssl=true"
clientOpts := options.Client().ApplyURI(highProxyUri)
clientOpts = clientOpts.SetTLSConfig(&tls.Config {
InsecureSkipVerify: true,
}).SetDirect(false).SetAuth(credential)
//URI of the direct connection. Note that SetDirect is set to true.
//directUri := "mongodb://host:8635/?ssl=true"
//clientOpts := options.Client().ApplyURI(highProxyUri)
//clientOpts = clientOpts.SetTLSConfig(&tls.Config {
// InsecureSkipVerify: true,
```



```
//}).SetDirect(true).SetAuth(credential)
// Connect to an instance.
ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
defer cancel()
client, err := mongo.Connect(ctx, clientOpts)
if err != nil {
   fmt.Println("Failed to connect to the mongo instance:", err)
   return
}
//Ping the primary node.
ctx, cancel = context.WithTimeout(context.Background(), 2*time.Second)
defer cancel()err = client.Ping(ctx, readpref.Primary())
if err != nil {fmt.Println ("Failed to ping the primary node: ",err)
   return
}
//Select a database and collection.
collection := client.Database("test").Collection("numbers")
//Insert a record.
ctx, cancel = context.WithTimeout(context.Background(), 5*time.Second)
defer cancel()
oneRes, err := collection.InsertOne(ctx, bson.D{{"name", "e"}, {"value", 2.718}})
if err != nil{fmt.Println("Failed to insert a record: ",err)
   return
}else {
   fmt.Println(oneRes)
}
Import {0} records in batches.
ctx, cancel = context.WithTimeout(context.Background(), 100*time.Second)
defer cancel()
docs := make([]interface{}, 100)
for i := 0; i < 100; i++{
   docs[i] = bson.D{{"name", "name"+strconv.Itoa(i)}, {"value", i}}
}
manyRes, err := collection.InsertMany(ctx, docs)
if err != nil {
   fmt.Println("Batch insertion failed: ",err)
   return
}else {
   fmt.Println(manyRes)
}
```

- Disabling SSL

```
//HA connection. The value of readPreference is primary (by default), indicating that the primary node is read-only. The value primaryPreferred indicates that the primary node is preferred. If the primary node is unavailable, the secondary node is read.
// Secondary: read-only node. If the secondary node is unavailable, an error is reported.
secondaryPreferred: The secondary node is preferred. If the secondary node is unavailable, the primary node is read.
// There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
// In this example, the username and password are stored in the environment variables. Before running this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
username = System.getenv("EXAMPLE_USERNAME_ENV")
password = System.getenv("EXAMPLE_PASSWORD_ENV")
highProxyUri := fmt.Sprintf("mongodb://%v:%v@host1:8635,host2:8635/?authSource=admin&replicaSet=replica&readPreference=secondaryPreferred",username,password)
ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
defer cancel()
clientOpts := options.Client().ApplyURI(highProxyUri)
client, err := mongo.Connect(ctx, clientOpts)
//Ping the primary node.
ctx, cancel = context.WithTimeout(context.Background(), 2*time.Second)
defer cancel()err = client.Ping(ctx, readpref.Primary())
if err != nil {
   fmt.Println("Failed to ping the primary node: ",err)
   return
}
//Select a database and collection.
```



```
collection := client.Database("test").Collection("numbers")
//Insert a record.
ctx, cancel = context.WithTimeout(context.Background(), 5*time.Second)
defer cancel()
res, err := collection.InsertOne(ctx, bson.D{{"name", "e"}, {"value", 2.718}})
if err != nil{
   fmt.Println("Failed to insert a record: ",err)
   return
}else {
   fmt.Println(res)
}
```

5.4.3 Accessing a Database

Accessing a Database

If there is an initialized MongoClient instance, run the following command to access a database:

```
db:= client.Database("test")
```

Assessing a Collection

After obtaining a MongoDatabase instance, run the following command to obtain a collection:

```
coll := db.Collection("testCollection")
```

Creating a Collection

You can use the CreateCollection() method to create a collection and specify the attributes of the collection during its creation.

```
db:= client.Database("test")
ctx, cancel = context.WithTimeout(context.Background(), 5*time.Second)
defer cancel()
sizeInBytes := int64(200000)
testCollection := db.CreateCollection(ctx,"testCollection",&options.CreateCollectionOptions{SizeInBytes:
&sizeInBytes})
```

5.4.4 Complete Example

```
import (
"context"
"crypto/tls"
"fmt"
"strconv"
"time"

"go.mongodb.org/mongo-driver/bson"
"go.mongodb.org/mongo-driver/mongo"
"go.mongodb.org/mongo-driver/mongo/options"
"go.mongodb.org/mongo-driver/mongo/readpref"
)
func main() {
  //Construct an authentication credential.
    // There will be security risks if the username and password used for authentication are directly written
into code. Store the username and password in ciphertext in the configuration file or environment variables.
    // In this example, the username and password are stored in the environment variables. Before running
this example, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as
needed.
    username = System.getenv("EXAMPLE_USERNAME_ENV")
    password = System.getenv("EXAMPLE_PASSWORD_ENV")
```



```
  credential := options.Credential{
    AuthMechanism: "SCRAM-SHA-1",
    AuthSource:    "admin",
    Username:      username,
    Password:      password,
  }
 //HA URI. Note that SetDirect is set to false.
  highProxyUri := "mongodb://host1:8635,host2:8635/?ssl=true"
  clientOpts := options.Client().ApplyURI(highProxyUri)
  clientOpts = clientOpts.SetTLSConfig(&tls.Config {
  InsecureSkipVerify: true,
  }).SetDirect(false).SetAuth(credential)
 //URI of the direct connection. Note that SetDirect is set to true.
 //directUri := "mongodb://host:8635/?ssl=true"
 //clientOpts := options.Client().ApplyURI(highProxyUri)
 //clientOpts = clientOpts.SetTLSConfig(&tls.Config {
 // InsecureSkipVerify: true,
 //}).SetDirect(true).SetAuth(credential)
 // Connect to an instance.
  ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
  defer cancel()
  client, err := mongo.Connect(ctx, clientOpts)
  if err != nil {
     fmt.Println("Failed to connect to the mongo instance:", err)
     return
  }
 //Ping the primary node.
  ctx, cancel = context.WithTimeout(context.Background(), 2*time.Second)
  defer cancel()err = client.Ping(ctx, readpref.Primary())
  if err != nil {fmt.Println ("Failed to ping the primary node: ",err)
     return
  }
 //Select a database and collection.
  collection := client.Database("test").Collection("numbers")
 //Insert a record.
  ctx, cancel = context.WithTimeout(context.Background(), 5*time.Second)
  defer cancel()
  oneRes, err := collection.InsertOne(ctx, bson.D{{"name", "e"}, {"value", 2.718}})
  if err != nil{fmt.Println("Failed to insert a record: ",err)
    return
 }else {
    fmt.Println(oneRes)
 }
 // Import {0} records.
 ctx, cancel = context.WithTimeout(context.Background(), 100*time.Second)
 defer cancel()
 docs := make([]interface{}, 100)
 for i := 0; i < 100; i++{
    docs[i] = bson.D{{"name", "name"+strconv.Itoa(i)}, {"value", i}}
 }
 manyRes, err := collection.InsertMany(ctx, docs)
 if err != nil {
    fmt.Println("Batch insertion failed: ",err)
    return
 }else {
    fmt.Println(manyRes)
  }
}
```


6 Performance White Paper

6.1 Replica Sets

6.1.1 Test Method

This section describes performance testing of GeminiDB Mongo replica sets (version 4.0), including the test environment, procedure, and results.

Test Environment

- Region: CN-Hong Kong
- AZ: AZ1
- Elastic Cloud Server (ECS): s3.2xlarge.2 flavor with 8 vCPUs, 16 GB of memory, and CentOS 7.5 64-bit image
- Replica set specifications: All specifications described in **Table 6-1**

Table 6-1 Replica set specifications

Replica Set No.	DB Version	Specifications
Replica set 1	4.0	4 vCPUs | 16 GB
Replica set 2	4.0	8 vCPUs | 32 GB
Replica set 3	4.0	16 vCPUs | 64 GB
Replica set 4	4.0	32 vCPUs | 128 GB

Test Tool

YCSB is an open-source tool for testing performance of databases. In this test, YCSB 0.12.0 is used.

For details on how to use this tool, see **YCSB**.

Test Metrics

Operations per Second (OPS): operations executed by a database per second

Test Procedure

1. Configure the **workload** file.

   Set values for fields **readproportion**, **insertproportion**, and **updateproportion** in the **workload** file by referring to **Table 6-2**.

   Set a value for field **recordcount** in the **workload** file by referring to **Table 6-3**.

2. Use workload_s1 as an example. Run the following command to prepare test data:

   ./bin/ycsb load mongodb -s -P workloads/workload_s1 -p mongodb.url=mongodb://${userName}:${password}@${mongodIP}:${port},${mongodIP}:${port},${mongodIP}:${port}/ycsb?authSource=admin&replicaSet=replica&readPreference=secondary -threads ${threadNum} 1>workload_s1_load.result 2> workload_s1_load.log

3. Run the following command to test performance:

   ./bin/ycsb run mongodb -s -P workloads/workload_s1 -p mongodb.url=mongodb://${userName}:${password}@${mongodIP}:${port},${mongodIP}:${port},${mongodIP}:${port}/ycsb?authSource=admin&replicaSet=replica&readPreference=secondary -threads ${threadNum} -p maxexecutiontime=1800 1>workload_s1_run.result 2> workload_s1_run.log

   NOTE

   - **${mongodIP}** indicates private IP addresses of primary and secondary nodes in the GeminiDB Mongo replica set.
   - **${password}** is the administrator password of the GeminiDB Mongo replica set.
   - **${threadNum}** indicates concurrent threads for running the test. In this test, the number of concurrent threads is 128.

Test Models

- Workload model

Table 6-2 Workload models

No.	Workload model
S1	100% insert
S2	90% update, 10% read
S3	65% read, 25% insert, 10% update
S4	90% read, 5% insert, 5% update
S5	50% update, 50% read
S6	100% read

- Concurrent threads: 128
- Document model

  Use the default settings of YCSB: The size of each document is 1 KB, and the default index is **_id**.
- Preset data volume

  Two types of preset data volumes were used to test performance of replica sets of each type of specifications.

  The following table describes preset data volumes for different replica set specifications.

  Table 6-3 Preset data volumes

Replica Set No.	Specifications	Small Data Volume	Large Data Volume
Replica set 1	4 vCPUs | 16 GB	Storage space: 10 GB RecordCount: 10000000	Storage space: 100 GB RecordCount: 100000000
Replica set 2	8 vCPUs | 32 GB	Storage space: 10 GB RecordCount: 10000000	Storage space: 100 GB RecordCount: 100000000
Replica set 3	16 vCPUs | 64 GB	Storage space: 10 GB RecordCount: 10000000	Storage space: 100 GB RecordCount: 100000000
Replica set 4	32 vCPUs | 128 GB	Storage space: 10 GB RecordCount: 10000000	Storage space: 100 GB RecordCount: 100000000

- Data consistency model

  Weak consistency: If **Write Concern** is set to **{w : 1, j : false}**, a response is returned after data is written to disks on a single node. If the default setting is kept **Write Concern**, data is asynchronously stored on disks.

6.1.2 Test Data

- The OPS of instances with different specifications is tested using different service models and **10 million rows of preset data**. For details, see the content in bold in **Table 6-4**.

Table 6-4 Test data

Instance Specifications	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 64 GB	32 vCPUs | 128 GB
Service Model S1	35263	69490	75332	92332
Service Model S2	13399	26732	39276	50833
Service Model S3	19959	55442	79135	98611
Service Model S4	23338	75905	89156	101306
Service Model S5	16829	35224	60616	81019
Service Model S6	29286	64882	97377	121071

*__OPS__: indicates the operations executed by a database per second. OPS is short for Operations per Second.

*__Service Model No.__: indicates the service model sequence number. For details, see **Table 6-2**.

- For weak consistency, the OPS of replica sets of different specifications can be tested using different service models and **100 million rows of preset data**. For details, see numbers in bold in **Table 6-5**.

Table 6-5 Test data

Instance Specifications	4 vCPUs | 16 GB	8 vCPUs | 32 GB	16 vCPUs | 64 GB	32 vCPUs | 128 GB
Service Model S1	18448	27976	33296	51920

Service Model S2	9004	14565	19961	26589
Service Model S3	15431	27476	32161	50332
Service Model S4	14926	25745	33258	48924
Service Model S5	11541	14272	17733	23160
Service Model S6	15637	17452	27827	49384

*__OPS__: indicates the operations executed by a database per second. OPS is short for Operations per Second.

*__Service Model No.__: indicates the service model sequence number. For details, see **Table 6-2**.

7 FAQs

7.1 Product Consulting

7.1.1 What Should I Pay Attention to When Using GeminiDB Mongo API?

1. DB instance operating systems (OSs) are invisible to you. Your applications can access a database only through an IP address and a port.
2. The backup files stored in OBS and the system containers used by GeminiDB API are invisible to you. They are visible only in the background GeminiDB Mongo API management system.
3. Precautions after purchasing DB instances:

   After purchasing DB instances, you do not need to perform basic database O&M operations, such as applying HA and security patches, but you should still note:

   a. The CPU, input/output operations per second (IOPS), and space are insufficient for the DB instances.
   b. The DB instance has performance problems and whether optimization is required.

7.1.2 What Is the Availability of GeminiDB Mongo Instances?

The formula for calculating the instance availability is as follows:

Instance availability = (1 – Failure duration/Total service duration) × 100%

The failure duration refers to the total duration of faults that occur during the running of an instance after you buy the instance. The total service duration refers to the total running time of the instance.

7.2 Permissions

7.2.1 How Do I Create a Read-only User for My GeminiDB Mongo Instance?

This section describes how to create a read-only user for a GeminiDB Mongo instance.

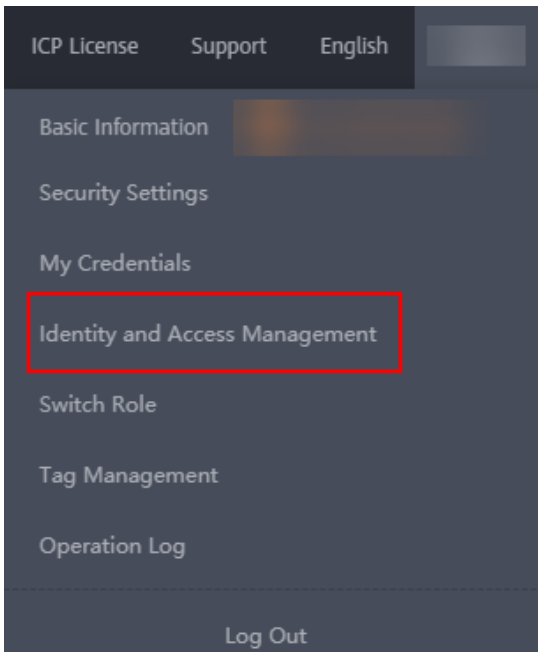
Step 1: Create a User Group and Grant Permissions

A user group is a collection of users. With IAM, you can create users, add them to a specific user group, and grant permissions to the user group. Then all users in the group inherit the permissions from the group. To create a user group and assign the permission to it, perform the following steps:

Step 1 Log in to Huawei Cloud using your HUAWEI ID.

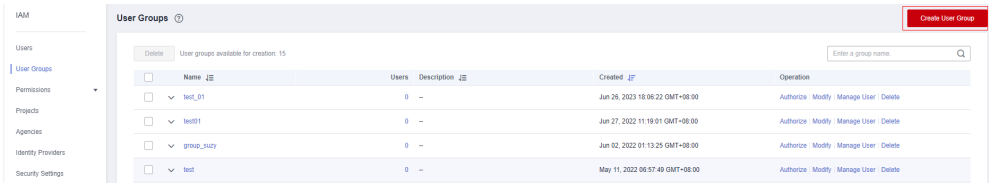
Step 2 On the management console, click the username in the upper right corner and then choose **Identity and Access Management**.

Figure 7-1 Choosing IAM



Step 3 On the IAM console, choose **User Groups** in the navigation pane. Then click **Create User Group**.

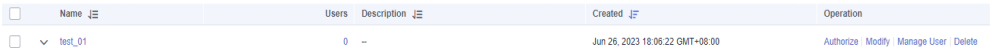
Figure 7-2 Creating a user group



Step 4 In the **Create User Group** dialog box, enter a user group name, for example, **test_01**, and click **OK**.

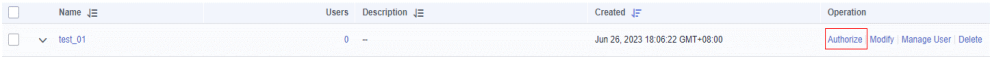
After the user group is created, you are redirected to the user group list page. The created user group is displayed in the list.

Figure 7-3 User group you created



Step 5 On the **User Groups** page, select the target group, and click **Authorize** in the **Operation** column.

Figure 7-4 Authorization



Step 6 Authorize the user group.

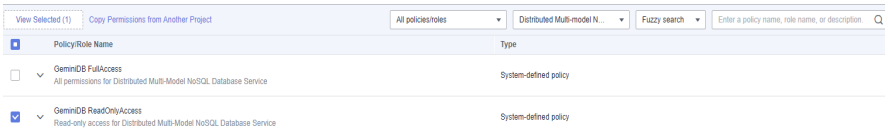
1. Select **Distributed Multi-model NoSQL Database Service**, enter **NoSQL** in the text box, and click the search icon.

   Figure 7-5 Selecting a service

2. In the permission list, select **GeminiDB ReadOnlyAccess** and click **OK**.

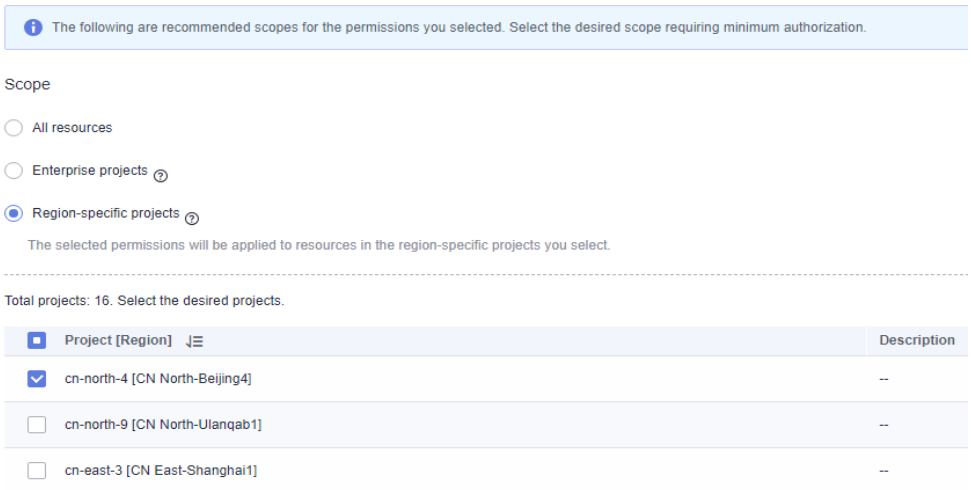
   Figure 7-6 Authorization

3. Select **Region-specific projects** for **Scope**.

   The authorization setting takes effect in only selected regions. If you hope that the setting takes effect in all regions, select **All projects**.

Figure 7-7 Setting the minimum authorization scope



4. After the authorization is successful, choose **Permissions** > **Authorization** to view authorization records of the user group.

Figure 7-8 Viewing permissions

----End

Step 2: Create an IAM User

IAM users can be created for employees or applications of an enterprise. Each IAM user has their own security credentials, and inherits permissions from the groups it is a member of. To create an IAM user, perform the following steps:

Step 1 On the IAM console, choose **Users** in the navigation pane. Then click **Create User**.

Step 2 Specify user information on the **Create User** page. To create more users, click **Add User**. You can add a maximum of 10 users at a time.

Figure 7-9 Creating a user

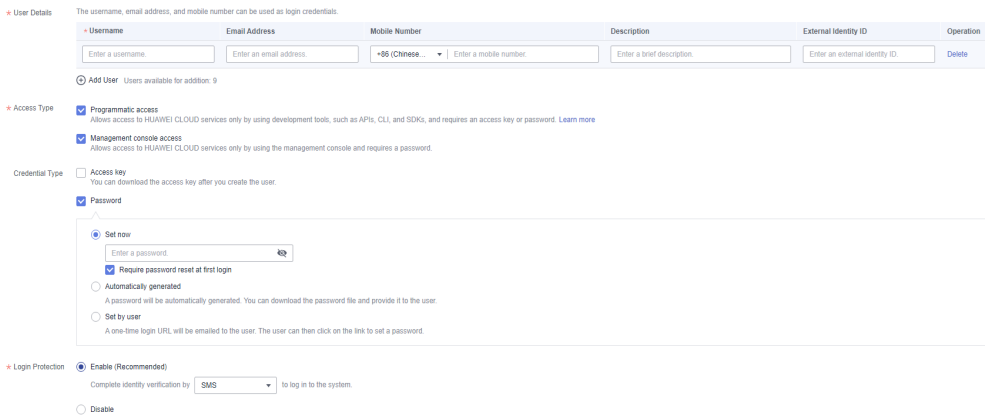
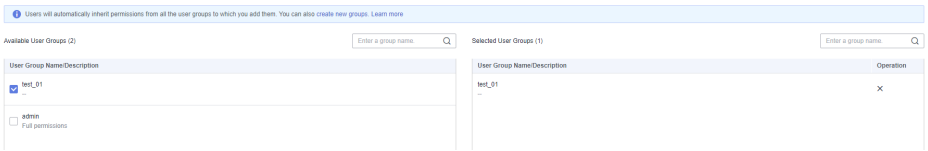


Table 7-1 Configuration items

Parameter	Description
User Details	• **Username**: Used for logging in to Huawei Cloud. For example, enter **James**. • **Email Address**: Email address bound to the IAM user. This parameter is mandatory if you choose **Credential Type** > **Password** > **Require password reset at first login**. • (Optional) **Mobile Number**: Mobile number bound to the IAM user. • (Optional) **Description**: Description of the user. • (Optional) **External Identity ID**: Identifies an enterprise user in federated login using SSO.
Access Type	• **Programmatic access**: After creation, you can download **access keys** of all the users you just created. • **Management console access**: The users you created can use accounts and passwords to log in to the Huawei Cloud console.
Credential Type	• **Access Key**: You can download the access key after you create the user. • **Password**: If you create multiple users at a time, set a password for each user and determine whether to require the users to reset passwords at first login. The **Automatically generated** option is not supported. If you create only one user, you can select any option of the above.
Login Protection	To ensure account security, you are advised to select **Enable**. To disable this function, see **Login Protection**.

Step 3 (Optional) Click **Next** and then **OK**.

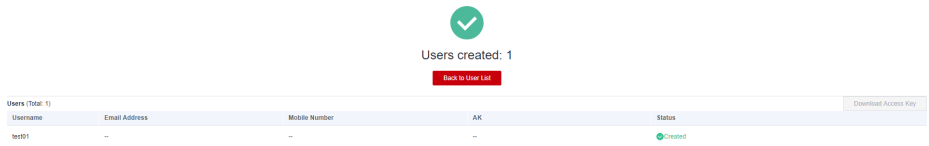
Figure 7-10 Adding the user to a user group



- Select user group **test_01**. You can add a user to one or more user groups. Then the user will inherit the permissions granted to these user groups. The default user group is **admin**, which has the administrator permissions and all of the permissions required to use all cloud resources.
- You can also create new groups as required.
- If you do not want to grant permissions to the user in this step, you can grant permissions by referring to **Step 5** later.

Step 4 Check the created user in the user list. If **Programmatic access** is selected, you can download the access key. You can also manage **access keys** on the **My Credentials** page.

Figure 7-11 Viewing the creation results

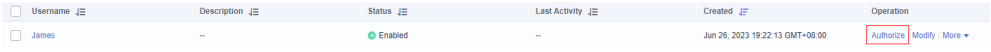


Step 5 If the new user is not authorized in **Step 3**, perform the following operations to authorize it.

In the user list, locate the created user, click **Authorize** in the **Operation** column, select the user group created in **Step 1: Create a User Group and Grant Permissions**, and click **OK**.

The user will inherit the permissions of the user group. This process is called authorization.

Figure 7-12 Authorization



----End

Step 3: Log In and Verify Permissions

After a user is created, use the user's username and credential to log in to Huawei Cloud, and verify that the user has the permissions defined by the **GeminiDB ReadOnlyAccess** policy. For more login methods, see **Signing In to Huawei Cloud**.

Step 1 On the Huawei Cloud login page, click **IAM User**.

Step 2 Enter the account name, username, and password, and click **Log In**.

- The account name is the name of the HUAWEI CLOUD account that created the IAM user.
- The username and password used here are the ones you enter when you created the IAM user.

If the login fails, contact the entity owning the account to verify the username and password. Alternatively, you can reset the password by referring to **How Do I Reset My Password?**.

Step 3 On the Huawei Cloud console, switch to the region where the user has been granted permissions.

Figure 7-13 Selecting the target region

Step 4 Choose **Service List** > **GeminiDB Mongo API**. Then click **Buy DB Instance** in the upper right corner. If a message appears indicating insufficient permissions, the **GeminiDB ReadOnlyAccess** policy has already taken effect.

Step 5 Choose any other service in the **Service List**. If a message appears indicating insufficient permissions, the **GeminiDB ReadOnlyAccess** policy has already taken effect.

----End

7.3 Billing

7.3.1 What Are the Differences Between Yearly/Monthly and Pay-per-use Billing Mode?

Yearly/Monthly is a prepaid billing mode in which resources are billed based on the service duration. This cost-effective mode is ideal when the duration of resource usage is predictable. It is recommended for long-term users.

Pay-per-use is a post payment mode, so you can start or stop an instance at any time. Pricing is listed on a per-hour basis, but bills are calculated based on the actual usage duration.

7.3.2 Can I Switch Between Yearly/Monthly and Pay-per-Use Payments?

You can change the billing mode from yearly/monthly to pay-per-use or vice versa.

- If you want to change the billing mode from yearly/monthly to pay-per-use, see **Changing the Billing Mode from Yearly/Monthly to Pay-per-Use**.
- If you want to change the billing mode from pay-per-use to yearly/monthly, see **Changing the Billing Mode from Pay-per-Use to Yearly/Monthly**.

7.4 Database Usage

7.4.1 How Do I Create a GeminiDB Mongo Session?

GeminiDB Mongo API is a cloud-native NoSQL database compatible with MongoDB. For details about how to create a session on, see **MongoDB official documents**.

7.4.2 Can GeminiDB Mongo API Be Downgraded?

Not supported.

7.4.3 Does GeminiDB Mongo API Support Read-Only Databases?

Not supported.

7.4.4 Does GeminiDB Mongo API Support Oplog Subscription?

Not supported.

7.4.5 How Do I Troubleshoot High CPU Usage of a GeminiDB Mongo Instance?

Analyzing Current Queries

1. Connect to a GeminiDB Mongo instance using Mongo Shell.
2. Run the following command to view operations being performed on the instance:

   db.currentOp()

   Command output:

```
{
	"raw" : {
		"shard0001" : {
			"inprog" : [
				{
					"desc" : "StatisticsCollector",
					"threadId" : "140323686905600",
					"active" : true,
					"opid" : 9037713,
					"op" : "none",
					"ns" : "",
					"query" : {

					},
					"numYields" : 0,
					"locks" : {

					},
					"waitingForLock" : false,
					"lockStats" : {

					}
				},
				{
					"desc" : "conn2607",
					"threadId" : "140323415066368",
					"connectionId" : 2607,
					"client" : "xxx.xxx.xxx.xxx:xxx",
					"appName" : "MongoDB Shell",
```



```
                        "active" : true,
                        "opid" : 9039588,
                        "secs_running" : 0,
                        "microsecs_running" : NumberLong(63),
                        "op" : "command",
                        "ns" : "admin.",
                        "query" : {
                                "currentOp" : 1
                        },
                        "numYields" : 0,
                        "locks" : {

                        },
                        "waitingForLock" : false,
                        "lockStats" : {

                        }
                }
        ],
        "ok" : 1
},
...
}
```

NOTE

- **client** indicates the IP address of the client that sends the request.
- **opid** indicates the unique operation ID.
- **secs_running** indicates the elapsed time for execution, in seconds. If the returned value of this field is too large, check whether there is something wrong with the request.
- **microsecs_running** indicates the elapsed time for execution, in microseconds. If the returned value of this field is too large, check whether there is something wrong with the request.
- **op** indicates the operation type. The operations can be **query**, **insert**, **update**, **delete**, or **command**.
- **ns** indicates the target collection.

For details, see the **db.currentOp()** command in the **official document**.

3. Based on the command output, check whether there are requests that take prolonged periods to process.

   If the CPU usage is low during routine operation but increases during specific operations, analyze the requests that take an overlong time to execute.

   If an abnormal query is found, find the **opid** corresponding to the operation and run **db.killOp(*opid*)** to kill it.

Analyzing Slow Queries

Slow query profiling has been enabled for GeminiDB Mongo instances by default. The system automatically records any queries whose execution takes longer than 500 ms to the **system.profile** collection in the corresponding database. You can:

1. Connect to a GeminiDB Mongo instance using Mongo Shell.
2. Select a specific database (using the **test** database as an example):
   ```
   use test
   ```
3. Check whether slow SQL queries have been collected in **system.profile**.
   ```
   show collections;
   ```
   - If the command output includes **system.profile**, slow SQL queries have been generated. Go to the next step.


```
mongos> show collections
system.profile
test
```

- If the command output does not contain **system.profile**, no slow SQL queries have been generated, and slow query analysis is not required.

```
mongos> show collections
test
```

4. Check the slow query logs in the database.

   db.system.profile.find().pretty()

5. Analyze slow query logs to find the cause of the high CPU usage.

   The following is an example of a slow query log. The log shows a request that scanned the entire table, including 1,561,632 documents. No indexes are used for query.

```
{
	"op" : "query",
	"ns" : "taiyiDatabase.taiyiTables$10002e",
	"query" : {
		"find" : "taiyiTables",
		"filter" : {
			"filed19" : NumberLong("852605039766")
		},
		"shardVersion" : [
			Timestamp(1, 1048673),
			ObjectId("5da43185267ad9c374a72fd5")
		],
		"chunkId" : "10002e"
	},
	"keysExamined" : 0,
	"docsExamined" : 1561632,
	"cursorExhausted" : true,
	"numYield" : 12335,
	"locks" : {
		"Global" : {
			"acquireCount" : {
				"r" : NumberLong(24672)
			}
		},
		"Database" : {
			"acquireCount" : {
				"r" : NumberLong(12336)
			}
		},
		"Collection" : {
			"acquireCount" : {
				"r" : NumberLong(12336)
			}
		}
	},
	"nreturned" : 0,
	"responseLength" : 157,
	"protocol" : "op_command",
	"millis" : 44480,
	"planSummary" : "COLLSCAN",
	"execStats" : {
	    "stage" :
"SHARDING_FILTER",
		[3/1955]
		"nReturned" : 0,
```



```
            "executionTimeMillisEstimate" : 43701,
            "works" : 1561634,
            "advanced" : 0,
            "needTime" : 1561633,
            "needYield" : 0,
            "saveState" : 12335,
            "restoreState" : 12335,
            "isEOF" : 1,
            "invalidates" : 0,
            "chunkSkips" : 0,
            "inputStage" : {
                  "stage" : "COLLSCAN",
                  "filter" : {
                        "filed19" : {
                              "$eq" : NumberLong("852605039766")
                        }
                  },
                  "nReturned" : 0,
                  "executionTimeMillisEstimate" : 43590,
                  "works" : 1561634,
                  "advanced" : 0,
                  "needTime" : 1561633,
                  "needYield" : 0,
                  "saveState" : 12335,
                  "restoreState" : 12335,
                  "isEOF" : 1,
                  "invalidates" : 0,
                  "direction" : "forward",
                  "docsExamined" : 1561632
            }
      },
      "ts" : ISODate("2019-10-14T10:49:52.780Z"),
      "client" : "xxx.xxx.xxx.xxx",
      "appName" : "MongoDB Shell",
      "allUsers" : [
            {
                  "user" : "__system",
                  "db" : "local"
            }
      ],
      "user" : "__system@local"
}
```


The following stages can be causes for a slow query:

- **COLLSCAN**, involving a full collection (full table) scan

  When a request (such as QUERY, UPDATE, and DELETE) requires a full table scan, a large amount of CPU resources are occupied. If you find **COLLSCAN** in the slow query log, CPU resources may be occupied.

  If such requests are frequent, create indexes for the fields to be queried.

- **docsExamined**, involving a full collection (full table) scan

  You can view the value of **docsExamined** to see how many documents are scanned in a query. A larger value indicates a higher CPU usage.

- **IXSCAN** and **keysExamined** scan indexes

  - An excessive number of indexes can affect the write and update performance.
  - If your application has more write operations, creating indexes may increase write latency.

  You can view the value of **keyExamined** to see how many indexes are scanned in a query. A larger value indicates a higher CPU usage.

If an index is inappropriate or it has many matching results, the CPU usage does not decrease greatly but the execution is slow.

Example: For the data of a collection, field **a** has few values (only values **1** and **2**), but field **b** has many values.

```
{ a: 1, b: 1 }
{ a: 1, b: 2 }
{ a: 1, b: 3 }
......
{ a: 1, b: 100000}
{ a: 2, b: 1 }
{ a: 2, b: 2 }
{ a: 2, b: 3 }
......
{ a: 1, y: 100000}
```

The following shows how to perform the {a: 1, b: 2} query.

db.createIndex({a: 1}): The query is slow because the **a** field has too many same values.

db.createIndex({a: 1, b: 1}): The query is slow because the **a** field has too many same values.

db.createIndex({b: 1}): The query is fast because the **b** field has a few same values.

db.createIndex({b: 1, a: 1}): The query is fast because the **b** field has a few same values.

For the differences between {a: 1} and {b: 1, a: 1}, see the **official documents**.

- **SORT** and **hasSortStage**, which may involve sorting a large amount of data.

  If the value of the **hasSortStage** parameter in the **system.profile** collection is **true**, the query request involves sorting. If the sorting cannot be implemented through indexes, the query results are sorted, and sorting is a CPU-intensive operation. When this happens, you need to create indexes for the fields that are frequently sorted.

  If the **system.profile** collection contains **SORT**, you can use indexing to improve sorting speed.

  Other operations, such as index creation and aggregation (combinations of traversal, query, update, and sorting), also apply to the preceding scenarios because they are also CPU-intensive operations. For more information about profiling, see the **official documents**.

Analyzing Service Capabilities

After analyzing and optimizing current requests and slow queries, all requests use appropriate indexes, and the CPU usage becomes stable. If the CPU usage remains high after the preceding troubleshooting operations are performed, the current

instance may be experiencing a performance bottleneck and cannot meet workload requirements. To address this issue, do as follows:

1. View monitoring information to analyze instance resource usage. For details, see **Viewing Metrics**.
2. Change the instance class or add shard nodes. For details, see the following documents based on the instance type.
   - **Adding Nodes**
   - **Changing Specifications of an Instance**

7.5 Client Installation

7.5.1 How Can I Install a MongoDB Client?

MongoDB official website provides client installation packages for different OSs. Download the official binary installation package at **https://www.mongodb.com/download-center#community**.

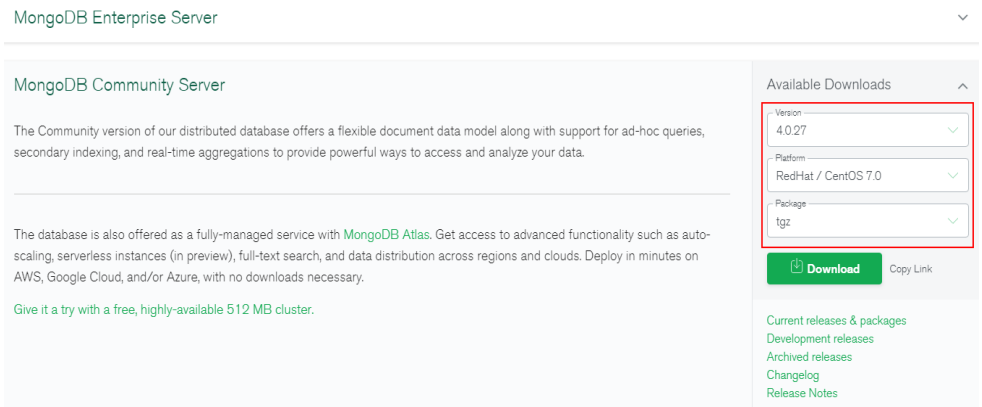
The following uses **RedHat/CentOS 7.0** and MongoDB 4.0.27 as examples to describe how to obtain the required installation package and install the MongoDB client. Select a client version based on the OS version of your ECS and the version of your GeminiDB Mongo instance.

Procedure

Step 1 Obtain the installation package.

1. Log in to **https://www.mongodb.com/download-center#community**.
2. Select **4.0.27** under **Version**, **RedHat/CentOS 7.0** for **Platform**, and **tgz** for **Package**. Make sure that the platform is consistent with the OS version of your ECS. **Figure 7-14** shows an example.

   Figure 7-14 MongoDB official web page



3. Click **Download** to download the binary installation package of version 4.0.27. The name of the installation package is **mongodb-linux-x86_64-rhel70-4.0.27.tgz**.

Step 2 Upload the installation package to the ECS.

You have created and logged in to an ECS. For details, see **Purchasing an ECS** and **Logging In to an ECS**.

Step 3 Decompress the installation package on the ECS.

tar zxvf mongodb-linux-x86_64-rhel70-4.0.27.tgz

Step 4 Obtain the client tool from the **bin** directory of the installation package.

cd mongodb-linux-x86_64-rhel70-4.0.27/bin

The common tools are as follows:

- MongoDB client mongo
- Data export tool mongoexport
- Data import tool mongoimport

Step 5 Before using a client tool, assign the execute permission to it.

- Run the **chmod +x mongo** command to grant a client permission to connect to a DB instance.
- Run the **chmod +x mongoexport** command to grant a client permission to export data.
- Run the **chmod +x mongoimport** command to grant a client permission to import data.

Step 6 Connect to a DB instance through the installed client.

For details about how to connect to a replica set instance, see **Connecting to a Replica Set Instance over a Private Network**.

When you run the MongoDB client, if an error similar to **Version libcrypto.so.10 not found** is displayed, check whether your client is compatible with the ECS OS.

----End

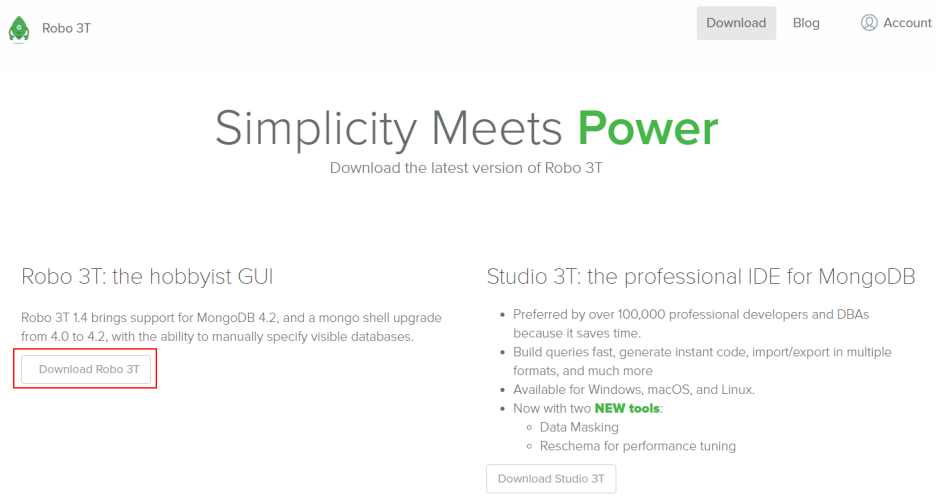
7.5.2 How Do I Install Robo 3T?

This section describes how to obtain the Robo 3T installation package and install Robo 3T.

Procedure

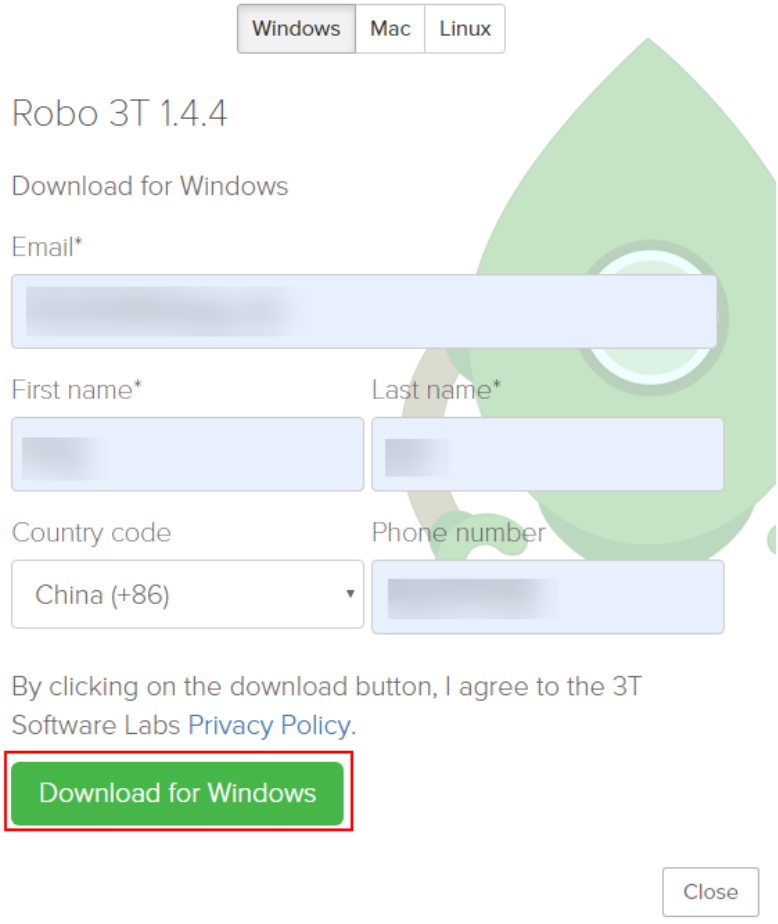
Step 1 Access the Robo 3T download address **https://robomongo.org/download** and click **Download Robo Only**.

Figure 7-15 Downloading page



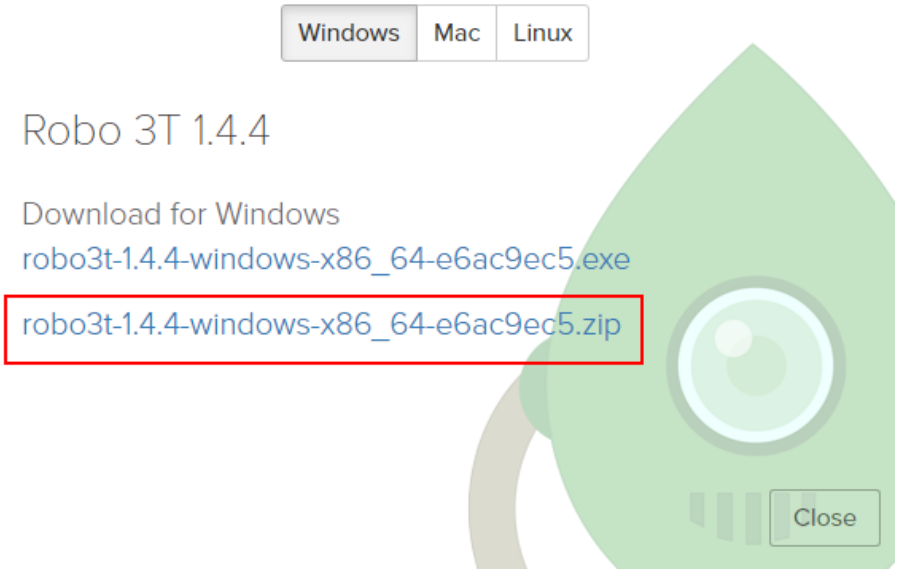
Step 2 In the dialog box that is displayed, enter required information and click **Download for Windows**.

Figure 7-16 Downloading Robo 3T



Step 3 Download **robo3t-1.4.4-windows-x86_64-e6ac9ec5.zip**.

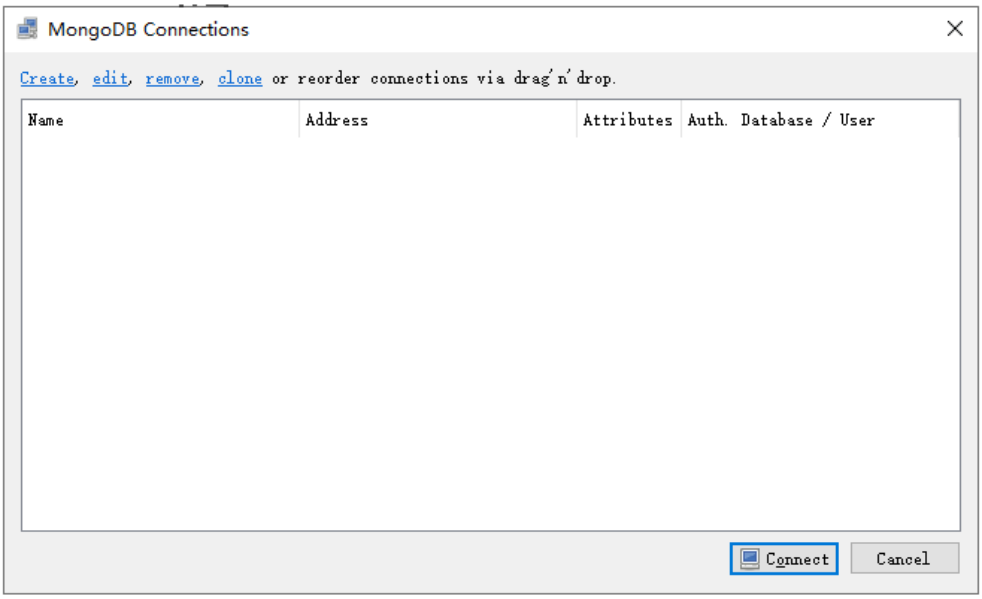
Figure 7-17 Downloading required software package



Step 4 Decompress the downloaded package obtained in **Step 3** and double-click the **robo3t.exe** file in the decompressed directory to start the installation.

Step 5 After the installation is complete, start the tool.

Figure 7-18 Main window



Step 6 Connect to an instance.

For details on how to connect to a replica set instance, see **Connecting to a Replica Set Instance Using Robo 3T**.

----End

7.6 Database Connection

7.6.1 How Can I Create and Connect to an ECS?

1. To create an ECS, see *Elastic Cloud Server User Guide*.
   - The ECS to be created must be in the same VPC with the GeminiDB Mongo instance to which it connects.
   - Configure the security group rules to allow the ECS to access to the instance.
2. To connect to an ECS, see "Logging in to an ECS" *Getting Started with Elastic Cloud Server User Guide*.

7.6.2 Can I Change the VPC of a GeminiDB Mongo Instance?

After a GeminiDB Mongo instance is created, the VPC where the instance resides cannot be changed.

However, you can change a VPC by restoring the full backup of your instance to the VPC you want to use. For details, see **Restoring Data to a New or Existing Instance**.

7.7 Backup and Restoration

7.7.1 How Long Does a GeminiDB Mongo Instance Store Backup Data For?

Automated backup data is kept based on the backup retention period you specified. There is no limit for the manual backup retention period. You can delete manual backups as needed.

7.8 Regions and AZs

7.8.1 What Is AZ and How Can I Select an AZ?

AZ

An AZ is a part of a physical region with its own independent power supply and network. An AZ is generally an independent physical equipment room, ensuring independence of the AZ.

Each region contains multiple AZs. If one AZ becomes faulty, the other AZs in the same region can continue to provide services normally.

AZs in the same region can communicate with each other over an intranet.

Selecting an AZ

You can deploy your instances in different AZs for high availability. If one of an AZ becomes faulty, databases in other AZs will not be affected. When selecting AZs:

- If only one AZ is available in a region, there is only one AZ in the region.
- The AZ of a purchased DB instance cannot be changed.
- The AZs in one region can communicate with each other over an intranet.

For more information, see **Regions and AZs**.

7.8.2 Can Different AZs Communicate with Each Other?

An AZ is a part of a physical region with its own independent power supply and network. An AZ is generally an independent physical equipment room, ensuring independence of the AZ.

Each region contains multiple AZs. If one AZ becomes faulty, the other AZs in the same region can continue to provide services normally.

By default, different AZs in the same VPC can communicate with each other through an internal network.

For more information, see **Regions and AZs**.

7.8.3 Can I Change the Region of a GeminiDB Mongo Instance?

No. After an instance is created, its region cannot be changed.

7.9 Instance Freezing, Release, Deletion, and Unsubscription

Why Are My GeminiDB Mongo Instances Released?

If your subscriptions have expired but not been renewed, or you are in arrears due to insufficient balance, your instances enter a grace period. If you do not renew the subscriptions or top up your account after the grace period expires, your instances will enter a retention period and become unavailable. If you still do not renew them or top up your account after the retention period ends, your instances will be released and your data stored will be deleted. For details, see **Resource Suspension and Release**.

Why Are My GeminiDB Mongo Instances Frozen?

Your instances may be frozen for a variety of reasons. The most common reason is that you are in arrears.

Can I Still Back Up Data If My Instances Are Frozen?

No. If your instances are frozen because your account is in arrears, go to top up your account to unfreeze your instances and then back up instance data.

How Do I Unfreeze My Instances?

If your instances are frozen because your account is in arrears, you can unfreeze them by renewing them or topping up your account. The frozen instances can be renewed, released, or deleted. Yearly/Monthly instances that have expired cannot be unsubscribed from, while those that have not expired can be unsubscribed from.

What Impacts Does Instance Freezing, Unfreezing or Release Have on My Services?

- After an instance is frozen:
  - It cannot be accessed, and your services will be interrupted. For example, if a GeminiDB Mongo instance is frozen, it cannot be connected.
  - No changes can be performed on it if it is a yearly/monthly instance.
  - It can be unsubscribed from or deleted manually.
- After it is unfrozen, you can connect to it again.
- Releasing an instance means deleting it. Before the deletion, GeminiDB Mongo API determines whether to **move the instance to the recycle bin** based on the recycling policy you specified.

How Do I Renew My Instances?

After a yearly/monthly instance expires, you can renew it on the **Renewals** page. For details, see **Renewal Management**.

Can My Instances Be Recovered After They Are Released or Unsubscribed From?

If your instance is moved to the recycle bin after being deleted, you can recover it from the recycle bin by referring to **Recycling an Instance**. If the recycling policy is not enabled, you cannot recover it.

When you unsubscribe from an instance, confirm the instance information carefully. If you have unsubscribed from an instance by mistake, purchase a new one.

How Do I Delete a GeminiDB Mongo Instance?

- To delete a pay-per-use instance, see **Deleting a Pay-per-Use Instance**.
- To delete a yearly/monthly instance, see **Unsubscribing from a Yearly/Monthly DB Instance**.

A Change History

Release Date	Description
2023-08-30	This issue is the twenty-fourth official release. Added **Billing**.
2023-07-18	This issue is the twenty-third official release. Added **Getting Started with Common Practices**.
2023-06-30	This issue is the twenty-second official release. Added event monitoring description in **Event Monitoring**.
2023-04-28	This issue is the twenty-first official release. In **Instance Specifications**, added the description of instances, each with a vCPUs to memory ratio of 1:8.
2023-03-31	This issue is the twentieth official release. Added **Updating the OS of an Instance**.
2022-08-11	This issue is the nineteenth official release. Added FAQ **Instance Freezing, Release, Deletion, and Unsubscription**.
2022-01-27	This issue is the eighteenth official release. Added description of batch operations in **Renewing Instances**, **Changing the Billing Mode from Pay-per-Use to Yearly/Monthly**, **Changing the Billing Mode from Yearly/Monthly to Pay-per-Use**, and **Unsubscribing from a Yearly/Monthly DB Instance**.
2021-11-30	This issue is the seventeenth official release. Added the description of SSL connections for replica set instances in **Buying a Replica Set Instance**, **Configuring an SSL Connection**, and GeminiDB Mongo.

Release Date	Description
2021-06-30	This issue is the sixteenth official release. Adjusted the document structure.
2021-04-30	This issue is the fifteenth official release. Added the minimum storage capacity supported by instances of different specifications in **Instance Specifications**. Supported one-click alarm reporting in **Configuring Alarm Rules**.
2021-04-15	This issue is the fourteenth official release. Adjusted the document structure.
2021-01-30	This issue is the thirteenth official release. Supported changing instance class, adding nodes to an instance, and scaling up storage capacity.
2020-12-30	This issue is the twelfth official release. Supported access across CIDR blocks for GeminiDB Mongo replica set instances.
2020-10-30	This issue is the tenth official release. • Supported up to 20 tags per instance. • Allowed users to select at least 1 GB each time they scale up. • Supported restoring the GeminiDB Mongo instance to an existing instance. • Supported scaling down the specifications of GeminiDB Mongo instances. • Added restricted commands for GeminiDB Mongo. • Supported advanced O&M for GeminiDB Mongo.
2020-09-30	This issue is the ninth official release, which incorporates the following changes: Supported instance quota management. Supported error log for GeminiDB Mongo. Supported parameter template for GeminiDB Mongo.

Release Date	Description
2020-08-30	This issue is the eighth official release. Supported the following functions of GeminiDB Mongo: • Supported batch purchase of pay-per-use or yearly/monthly instances. • Allowed users to view slow query log details, enable original slow query log, and export log files. • Supported instance name modifications. • Supported instance recycle bin. • Supported resetting the administrator password for a replica set instance. • Supported changing the instance port for a replica set instance. • Supported changing the replica set instance security group.
2020-07-30	This issue is the seventh official release. Supported the following functions of GeminiDB Mongo: • Purchasing yearly/monthly instances • Renewing yearly/monthly instances • Changing the billing mode of instances from pay-per-use to yearly/monthly • Changing the billing mode of instances from yearly/monthly to pay-per-use • Unsubscribing from yearly/monthly instances • Scaling up storage space • Changing instance class • Binding and unbinding an EIP • Accessing instances over a public network • Restarting an instance • Managing connections
2020-06-30	This issue is the sixth official release. Added the following functions to GeminiDB Mongo: • Resetting a password • Configuring cross-AZ disaster recovery
2020-04-30	This issue is the fifth official release. Added monitoring metrics.
2020-03-13	This issue is the third official release. Supported slow query logs.
2019-11-18	This issue is the first official release.

